



Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- Pengutipan tidak merugikan kepentingan yang wajar Unand.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Unand.

# **RANCANGAN BANGUNA ALAT PENGONTROL BAN BERJALAN BERBASIS MIKROKONTROLER AT89S51 DENGAN MENGUNAKAN BAHASA C**

## **TESIS**



**JUSNIATI  
06214007**

**PROGRAM PASCASARJANA  
UNIVERSITAS ANDALAS  
2008**

***Rancang Bangun Alat Pengontrol Ban Berjalan  
Berbasis Mikrokontroler  
AT89S51 Dengan Menggunakan Bahasa C***

oleh : JUSNIATI

(Di bawah bimbingan Dr. Akrajas dan Drs. Wildian, M.si)

**RINGKASAN**

Di dalam pabrik-pabrik, industri, bandara, maupun supermarket banyak barang yang akan dipindahkan dari suatu tempat ke tempat lainnya. Untuk itu diperlukanlah sebuah alat yang dapat membantu memindahkan barang-barang tersebut, tentunya sebuah alat yang efektif dan efisien cara kerjanya, agar dapat membawa keuntungan dalam meningkatkan produktifitas kerja. Salah satu cara yang ditempuh adalah dengan menggunakan sistem kontrol ban berjalan berbasis mikrokontroler AT89S51.

Tujuan penelitian : 1) Merancang bangun alat pengontrol sistem ban berjalan berbasis mikrokontroler AT89S51. 2) Dengan penggunaan alat ini diharapkan dapat memudahkan pekerjaan manusia dan meningkatkan produktifitas kerja.

Penelitian ini dilakukan di Laboratorium Instrumentasi Dan Elektronika Unand Padang, sejak Juli 2007 sampai Juni 2008. Alat dan bahan yang digunakan adalah komponen elektronika dan bahan pendukung lainnya yang meliputi PC, *software Eagle*, *software Mide-51*, *software Isp*, multimeter digital, solder, kabel penghantar, PCB, larutan  $\text{FeCl}_3$ , setrika, slide transparan, timah, dan sumber arus

dari stop kontak 220 volt. Komponen elektronika yang digunakan meliputi: resistor, kapasitor, transistor, motor, mikrokontroler AT89S51, led, kristal, tombol(*switch*).

Model penelitian ini adalah penelitian eksperimen laboratorium. Pada penelitian ini ada dua tahap yang harus dilakukan yaitu merancang perangkat keras (*hardware*) dan merancang perangkat lunak (*software*). Rancang bangun perangkat keras meliputi: rancangan rangkain catu daya, rancangan rangkaian minimum mikrokontroler AT89S51, rancangan rangkaian *driver*, sedangkan rancangan perangkat lunak meliputi rancangan diagram alir dan rancangan program untuk menggerakkan ban berjalan ke kiri dan ke kanan serta mengatur kecepatan alat tersebut.

Tata laksana penelitian untuk perancangan perangkat keras : 1) Uji coba rangkaian dengan merangkai komponen pada breadboard. 2) Perancangan tata letak komponen dengan menggunakan software eagle. 3) Hasil perancangan disetrikakan pada PCB 4) Komponen kemudian disolder. 5) Uji keluaran per blok rangkaian. 6) Uji keluaran rangkaian secara keseluruhan. 7) Pembuatan perangkat lunak. 8) Pengujian akhir meliputi pengujian perangkat keras dan perangkat lunak. Tata laksana penelitian untuk perancangan perangkat lunak diawali dengan merancang diagram alir, diteruskan dengan merancang program dengan menggunakan bahasa pemograman C. Untuk menulis program digunakan *software Mide-51*, dan untuk menanamkan program tersebut ke dalam mikrokontroler digunakan perangkat lunak *Isp*. Dari hasil penelitian ini diperoleh sebuah ban berjalan yang dapat bergerak dalam dua arah yang berlawanan (ke kiri dan ke kanan) dan mempunyai dua pilihan untuk kecepatan masing-masing 1,046 putaran per sekon dan 0.6906 putaran per sekon. Berat beban maksimum yang dapat dipindahkan oleh ban berjalan ini adalah 220 gram serta efisiensi kerja alat adalah sebesar 45,83 %.

## PERNYATAAN KEASLIAN TESIS

Saya menyatakan dengan sebenar-sebanarnya bahwa tulisan dalam tesis yang berjudul “ **RANCANG BANGUN BAN BERJALAN BERBASIS MIKROKONTROLER AT89S51** “ adalah hasil karya sendiri dan bukan merupakan jiplakan dari hasil karya orang lain, kecuali kutipan yang sumbernya dicantumkan. Jika dikemudian hari pernyataan ini tidak benar, maka status kelulusan dan gelar yang saya peroleh menjadi batal dengan sendirinya.

Padang, 14 Juli 2008

Yang Membuat Pernyataan

Dra. Jusniati





## RIWAYAT HIDUP

Penulis dilahirkan pada tanggal 7 November 1967 di Pariaman sebagai anak ketiga dari ayah Alihar BA dan ibu Aisyah. Penulis menamatkan Sekolah Dasar di SD Bungo Tanjung Pariaman tahun 1980, menamatkan Sekolah Menengah Pertama di SMP 3 Pariaman tahun 1983, tahun 1986 tamat di SMA 1 Pariaman, dan tahun 1990 memperoleh gelar sarjana di Program Studi Pendidikan Fisika FPMIPA IKIP Padang.

Pada tahun 1994 penulis menikah dengan Drs Aljufri dan dikaruniai tiga orang anak yaitu : Intan, Hafiz, dan Fadel. Sejak tahun 1992 sampai sekarang penulis ditugaskan sebagai seorang guru fisika di SMA 1 VII Koto Sungai Sarik , kabupaten Padang Pariaman. Pada tahun 2006 memperoleh kesempatan meneruskan pendidikan pada program Pascasarjana Universitas Andalas di Padang



## KATA PENGANTAR

Puji syukur penulis panjatkan kehadiran Allah SWT, yang telah melimpahkan hidayahNya hingga penulis dapat menyelesaikan tesis ini. Tesis ini ditulis berdasarkan hasil penelitian yang berjudul "Rancang Bangun Alat Pengontrol Ban Berjalan Berbasis Mikrokontroler AT89S51 Dengan Menggunakan Bahasa C"

Dalam penyelesaian tesis ini penulis menyampaikan terima kasih kepada Bapak DR. Akrajas selaku ketua komisi pembimbing, Bapak Drs. Wildian, M.Si selaku anggota komisi pembimbing dan juga sebagai Ketua Jurusan Fisika FMIPA Universitas Andalas, Bapak/ibu staf pengajar di jurusan Fisika, Bapak Prof. Dr. Ir. H. Novirman Jamarun, M.Sc sebagai Direktur Program Pascasarjana beserta karyawan dan karyawan. Rekan-rekan uni Hasnida, uni Tina, Denok, Wahid, Pak Yun, dan semua rekan di laboratorium, terimakasih untuk segalanya.

Akhirnya penulis berharap semoga tesis ini dapat bermanfaat dalam mengembangkan ilmu pengetahuan khususnya di bidang ilmu pengetahuan Instrumentasi dan Elektronika. Sebagai manusia biasa dalam penulisan tesis ini penulis tidak luput dari kesalahan dan kekhilafan. Untuk itu penulis harapkan kritik dan saran yang membangun dari semua pihak. Atas perhatian penulis ucapkan terimakasih.

Padang, Juli 2008

Penulis

## DAFTAR ISI

	Halaman
<b>KATA PENGANTAR</b> .....	ix
<b>DAFTAR ISI</b> .....	x
<b>DAFTAR TABEL</b> .....	xii
<b>DAFTAR GAMBAR</b> .....	xii
<b>DAFTAR LAMPIRAN</b> .....	xiv
<b>BAB. I PENDAHULUAN</b>	
1.1 Latar Belakang .....	1
1.2 Tujuan dan Manfaat Penelitian .....	3
1.3 Batasan Masalah.....	3
<b>BAB. II TINJAUAN PUSTAKA</b>	
2.1 Mikrokontroler .....	4
2.2 Pengenalan Bahasa C .....	9
2.2.1 Bahasa C untuk MCS51 .....	9
2.2.2 Bentuk Umum Program C .....	9
2.3 Motor DC .....	10
2.4 Transistor Sebagai Penguat Arus .....	12
2.5 Relai .....	13
2.6 Hubungan roda-roda.....	14
2.6.1 Roda-roda Sepusat .....	15
2.6.2 Roda-roda Bersinggungan.....	15
2.6.3 Roda-roda yang dihubungkan dengan Sabuk.....	16

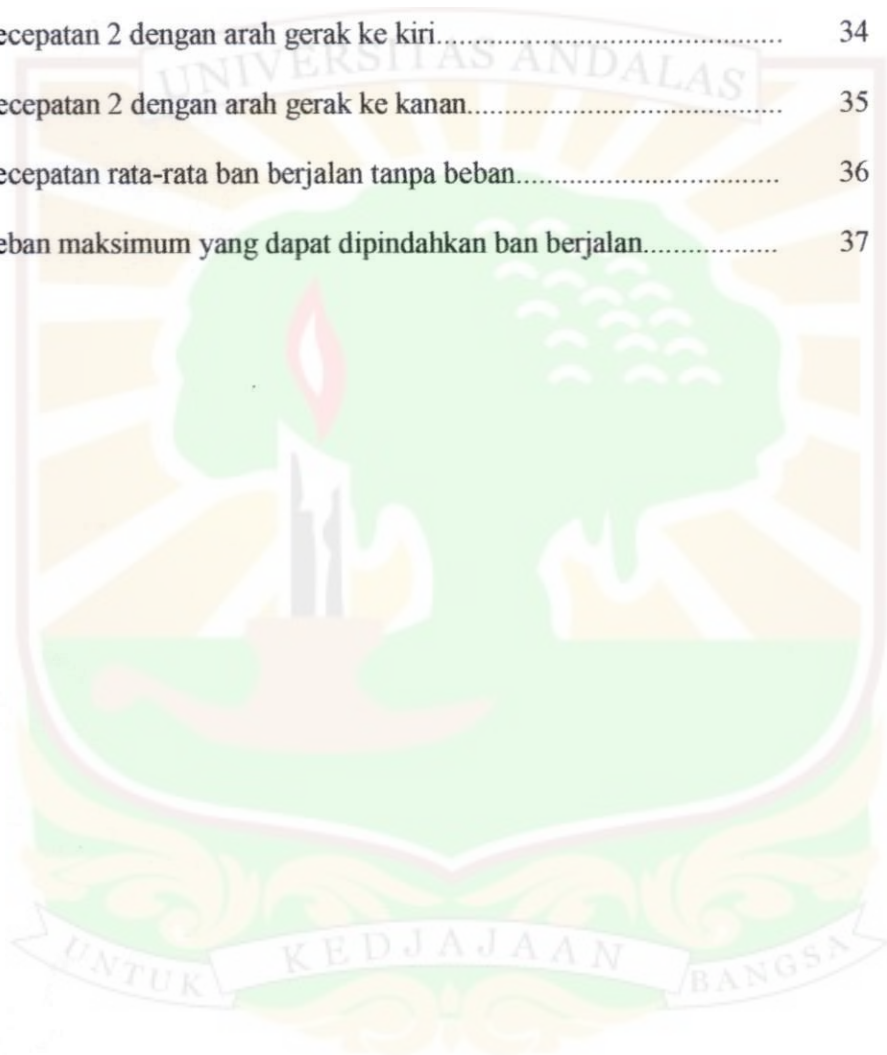


2.7 Perangkat Lunak Mikrokontroler AT89S51 .....	16
<b>BAB. III METODE PENELITIAN</b>	
3.1 Alat yang Digunakan Dalam Penelitian .....	20
3.2 Tata Laksana Penelitian .....	21
3.3 Rancang Bangun Alat Pengontrol Ban Berjalan .....	21
3.4 Rangkaian Catu Daya .....	22
3.5 Rangkaian Mikrokontroler AT89S51 .....	24
3.6 Rangkaian Driver .....	25
3.7 Rangkaian Pengatur Kecepatan .....	27
3.8 Rancangan Perangkat Lunak .....	28
<b>BAB. IV HASIL DAN PEMBAHASAN</b>	
4.1 Perangkat Keras .....	30
4.2 Hasil Pengujian .....	31
4.3 Data Hasil Pengukuran Kecepatan Ban Berjalan .....	32
4.4 Data Hasil pengukuran Beban Maksimum .....	36
4.5 Pembahasan .....	36
<b>BAB. V KESIMPULAN DAN SARAN</b> .....	38
<b>DAFTAR PUSTAKA</b> .....	39
<b>LAMPIRAN</b> .....	40



## DAFTAR TABEL

Nomor	halaman
1. Kecepatan 1 dengan arah .....	32
2. Kecepatan 1 dengan arah gerak ke kanan .....	33
3. Kecepatan 2 dengan arah gerak ke kiri.....	34
4. Kecepatan 2 dengan arah gerak ke kanan.....	35
5. Kecepatan rata-rata ban berjalan tanpa beban.....	36
6. Beban maksimum yang dapat dipindahkan ban berjalan.....	37



## DAFTAR GAMBAR

Nomor	halaman
1. Mikrokontroler AT 89S51.....	6
2. Konfigurasi pin mikrokontroler AT 89S51.....	7
3. Motor DC .....	10
4. Skema prinsip kerja motor DC .....	11
5. Transistor PNP .....	13
6. Relai .....	14
7. Roda-roda sepusat .....	15
8. Roda-roda yang bersinggungan .....	15
9. Roda-roda yang dihubungkan dengan sabuk .....	16
10. Diagram blok sistem kontrol alat .....	22
11. Rangkaian catu daya .....	23
12. Rangkaian target skematik mikrokontroler AT89S51.....	24
13. Rangkaian <i>driver</i> .....	26
14. Rangkaian relai .....	27
15. Diagram alir .....	28
16. Perangkat keras rancang bangun alat kontrol ban berjalan.....	30

## DAFTAR LAMPIRAN

Nomor	halaman
1. Tahap-tahap pembuatan rangkaian dengan menggunakan perangkat lunak Eagle.....	40
2. Bahasa pemograman C untuk mikrokontroler AT89S51.....	47
3. Tahap-tahap pemograman mikrokontroler dengan menggunakan perangkat lunak M-IDE51.....	50
4. Rangkaian lengkap alat kontrol ban berjalan berbasis mikrokontroler AT89S51.....	54
5. Data sheet mikrokontroler AT89S51.....	55



# BAB I

## PENDAHULUAN

Pada pendahuluan akan dijelaskan tentang latar belakang masalah, tujuan penelitian, manfaat penelitian serta pembatasan masalah dalam penelitian.

### 1.1 Latar Belakang

Ban berjalan (*belt conveyor*) banyak digunakan di dalam pabrik-pabrik, industri, bandara maupun supermarket untuk memindahkan barang-barang dari satu tempat ke tempat lain. Selain itu, ban berjalan juga digunakan pada peralatan olahraga (*treadmill*). Pemindahan barang-barang ini kurang efisien jika dikerjakan oleh manusia. Ban berjalan yang telah ada hanya dapat memindahkan barang dalam satu arah dan satu pilihan untuk kecepatan. Dengan ditemukannya teknologi mikrokontroler yang sering juga disebut *single chip mikrocomputer* maka ban berjalan dapat dikontrol arah dan kecepatannya sesuai dengan kebutuhan. Dengan demikian efisiensi kerja ban berjalan akan lebih tinggi dan mampu bekerja dalam waktu relatif lama tanpa pengawasan manusia. Dewasa ini mikrokontroler telah diproduksi dalam jumlah yang banyak, sehingga harganya jauh lebih murah.

Dengan alasan inilah penulis bermaksud mengerjakan tugas akhir merancang bangun alat pengontrol ban berjalan berbasis mikrokontroler AT89S51 dengan tujuan menghasilkan alat kontrol ban berjalan yang dapat bergerak dalam dua arah (ke kiri dan ke kanan) dan dua kecepatan agar



efisiensi kerja lebih tinggi yang sangat berguna untuk membantu pekerjaan manusia.

Penelitian tentang rancang bangun alat pengontrol ban berjalan yang penulis lakukan merupakan lanjutan penelitian yang telah dilakukan sebelumnya di Laboratorium Elektronika dan Instrumentasi Fisika Universitas Andalas yaitu rancang bangun sistem kontrol motor DC berbasis mikrokontroler AT89C2051 (Mardiah, 2005). Penelitian terdahulu yang dilakukan Ainun Mardiah adalah sistem kontrol yang digunakan dengan mikrokontroler AT89C2051 hanya dapat memutar arah motor DC ke kiri dan ke kanan dengan bahasa pemrogramannya *assembly*.

Kelebihan penelitian yang penulis lakukan adalah pengontrolan motor DC dengan menggunakan mikrokontroler AT89S51 dapat memutar arah motor DC ke kiri dan ke kanan dan putaran motor DC ini dimanfaatkan untuk menggerakkan suatu alat yang dinamakan ban berjalan. Selain itu pada penelitian ini kecepatan perputaran motor DC dapat diubah-ubah sehingga kecepatan ban berjalan itu dapat diubah-ubah sesuai dengan kebutuhan. Instruksi-instruksi yang digunakan untuk mengatur kerja ban berjalan itu ditulis dalam bahasa C.

## 1.2 Tujuan Penelitian dan manfaat Penelitian

Tujuan penelitian ini adalah merancang bangun alat pengontrol ban berjalan berbasis mikrokontroler **AT89S51** dengan menggunakan motor DC sebagai alat penggeraknya.

Dengan adanya penelitian ini diharapkan dapat memberikan manfaat kepada penulis untuk memahami prinsip kerja alat pengontrol sistem ban berjalan dengan menggunakan mikrokontroler **AT89S51**, dan kepada masyarakat agar dapat mempermudah pekerjaan dalam memindahkan barang dari suatu tempat ke tempat lain dari dua arah yang berlawanan dengan dua pilihan kecepatan yang tersedia sehingga efisiensi kerja alat lebih tinggi.

## 1.3 Batasan Masalah

Masalah pada penelitian ini dibatasi hanya pada perangkat keras serta perangkat lunak dari alat ban berjalan dengan menggunakan mikrokontroler **AT89S51** dan bahasa pemrograman C.

## BAB II

### TINJAUAN PUSTAKA

Pada bab ini akan dijelaskan kajian teori tentang : mikrokontroler, pengenalan bahasa C, motor DC, transistor, relai, hubungan roda-roda, perangkat lunak mikrokontroler AT89S51, dan pewaktuan mikrokontrol AT89S51.

#### 2.1 Mikrokontroler

Mikrokontroler adalah *Central Processing Unit* ( CPU ) yang disertai dengan memori serta sarana masukan / keluaran dan dibuat dalam bentuk sekeping *chip*, sehingga sering juga disebut *single chip microcomputer* (Budiono, 2000 ).

##### Deskripsi Mikrokontroler

Sebuah mikrokontroler memiliki *Central Processing Unit (CPU)*, bus yang meliputi bus alamat, bus data, bus kontrol dan memori yang terdiri dari *RAM*, dan *ROM*. CPU terdiri atas dua bagian, yaitu unit pengendali (*control unit*) serta unit aritmatika dan logika (*ALU*). Fungsi utama unit pengendali adalah mengambil, mengkodekan, dan melaksanakan urutan instruksi sebuah program yang tersimpan dalam memori. Unit pengendali menghasilkan dan mengatur sinyal pengendali yang diperlukan untuk menyerempakkan operasi, aliran dan instruksi program. Unit aritmatika dan logika berfungsi untuk melakukan proses perhitungan yang diperlukan selama program dijalankan



serta mempertimbangkan suatu kondisi dan mengambil keputusan yang diperlukan untuk instruksi-instruksi berikutnya. Bus alamat berfungsi sebagai sejumlah lintasan saluran pengalamatan antara alat dengan sebuah mikrokontroler. Pengalamatan ini harus ditentukan terlebih dahulu untuk menghindari terjadinya kesalahan pengiriman sebuah instruksi dan terjadinya bentrok antara dua buah alat yang bekerja secara bersamaan.

Bus data merupakan sejumlah lintasan saluran keluar masuknya data dalam suatu mikrokontroler. Pada umumnya saluran data yang masuk sama dengan saluran data yang keluar. Bus kontrol atau bus pengendali berfungsi untuk menyerempakkan operasi mikrokontroler dengan operasi rangkaian luar. Pada sebuah mikrokontroler terdapat sebuah memori yang berfungsi untuk menyimpan data atau program. Ada beberapa buah jenis memori, diantaranya **RAM (Random Acces Memory)** dan **ROM (Read Only Memory)**. RAM merupakan memori yang dapat dibaca dan ditulis. RAM biasanya digunakan untuk menyimpan data atau sering disebut dengan memori data saat program bekerja. Data yang ada pada RAM akan hilang bila catu daya dari RAM dimatikan sehingga RAM hanya dapat digunakan untuk menyimpan data sementara. ROM merupakan memori yang hanya dapat dibaca. Data yang disimpan di ROM tidak akan hilang meskipun catu daya dimatikan. Dari sifatnya itu maka ROM sering dipakai untuk menyimpan program. Ada beberapa jenis ROM diantaranya ROM, PROM, EPROM, dan EEPROM. ROM merupakan memori yang sudah diprogram oleh pabrik. PROM dapat diprogram oleh pemakai tetapi hanya dapat ditulis sekali saja. UV-EPROM merupakan PROM yang dapat diprogram atau ditulis beberapa



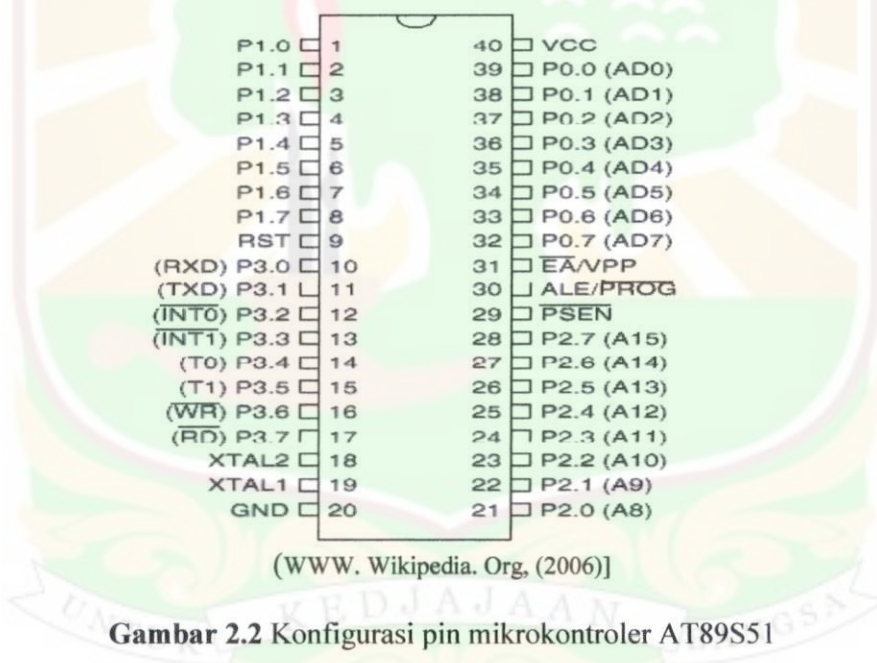
kali dan dapat dihapus dengan sinar ultraviolet. Flash PEROM adalah PROM yang dapat ditulis ulang beberapa kali dan dapat dihapus secara elektrik atau dengan tegangan listrik. UV-EPROM harganya lebih mahal dari Flash PEROM, karena itu Flash PEROM lebih populer dan diminati programmer mikrokontroler.

Mikrokontroler AT89S51 merupakan salah satu keluarga dari MCS-51 keluaran Atmel. Jenis mikrokontroler ini pada prinsipnya dapat digunakan untuk mengolah data per bit ataupun data 8 bit secara bersamaan. Sebuah mikrokontroler dapat bekerja bila di dalam mikrokontroler tersebut terdapat sebuah program yang berisi instruksi-instruksi yang akan digunakan untuk menjalankan sistem mikrokontroler tersebut. Instruksi-instruksi sebuah program dari setiap jenis mikrokontroler mempunyai beberapa perbedaan. Pada prinsipnya program pada mikrokontroler memiliki beberapa set instruksi dan tiap instruksi itu dijalankan secara bertahap atau berurutan. Bentuk fisik mikrokontroler AT89S51 dapat dilihat pada Gambar 2.1.



**Gambar 2.1** Mikrokontroler AT89S51

Beberapa fasilitas yang dimiliki oleh mikrokontroler AT89S51 adalah sebagai berikut : Sebuah *Central Processing Unit* 8 bit Osilator internal dan rangkaian pewaktu, RAM internal 128 byte, flash memori 4 kbyte, lima buah jalur interupsi (2 buah interupsi eksternal dan 3 buah interupsi internal), empat buah *programmable port* I/O yang masing-masing terdiri dari 8 buah jalur I/O. Sebuah port serial dengan kontrol serial *full duplex UART*, kemampuan untuk melaksanakan operasi aritmatika dan operasi logika, serta kecepatan dalam melaksanakan instruksi per siklus 1 mikrodetik pada frekuensi 12 MHz. Tata letak pin mikrokontroler AT89S51 terlihat pada Gambar 2.2.



Mikrokontroler AT89S51 terdiri atas 40 pin, di antaranya: pin 1 sampai 8 (port 1) adalah port paralel 8 bit 2 arah (*bidireksional*) dengan *internal pull-up* yang dapat digunakan untuk berbagai keperluan, seperti 4 masukan TTL dan sebagai saluran alamat saat pemrograman dan verifikasi,

pin 9 (*reset*) adalah masukan *reset* (aktif tinggi). Pulsa transisi dari rendah ke tinggi akan *mereset* AT89S51. Pin ini dihubungkan dengan rangkaian *power on reset* yang terdiri dari sebuah resistor dan kapasitor. Pin 10 sampai 17 (port 3) adalah port paralel 8 bit dua arah yang memiliki fungsi pengganti. Fungsi pengganti meliputi TxD (*Transmit Data*), RxD (*Receive data*), Int0 (*Interrupt 0*), Int1 (*Interrupt 1*), T0 (*timer 0*), T1 (*timer 1*), WR (*write*), dan RD (*read*). Bila fungsi pengganti tidak dipakai pin-pin ini dapat berguna sebagai port paralel 8 bit serbaguna. Pin 18 (*X-tal 2*) adalah pin keluaran rangkaian *inverting oscillator amplifier (output oscillator)*. Pin 19 (*X-tal 1*) adalah pin masukan ke rangkaian *inverting oscillator amplifier (input oscillator)*. Pin 20 (*ground*), dihubungkan ke *ground*. Pin 21 sampai 28 (Port 2) adalah port paralel 2 selebar 8 bit dua arah (*bidirectional*) yang berfungsi sebagai saluran/bus alamat tinggi. Port 2 ini mengirimkan byte alamat bila dilakukan pengaksesan memori eksternal. Pin 29 adalah pin PSEN (*Program Store Enable*) yang merupakan sinyal pengontrol yang membolehkan program memori eksternal masuk ke dalam bus selama proses pemberian atau pengambilan instruksi (*fetching*). Pin ini berfungsi pada saat pengeksekusi program yang terletak pada memori eksternal. Pin 30 adalah pin ALE (*Address Latch Enable*) yang digunakan untuk menahan alamat memori eksternal selama pelaksanaan instruksi. ALE hanya aktif pada saat mengakses memori eksternal (*Movx, Movc*).

Pin 31 adalah pin EA (*External Acces Enable*) yang berfungsi untuk mengeksekusi program dari memori internal jika dihubungkan ke Vcc dan memori eksternal jika dihubungkan ke *ground*. Bila diberi logika rendah



(*low*), mikrokontroler akan melaksanakan seluruh instruksi dari memori program luar. Pin 32 sampai 39 (Port 0) adalah port I/O 8 bit (*Open Drain*) dua arah. Bila digunakan untuk mengakses program luar, port ini akan *memultipleks* alamat memori dengan dataserta pin 40 (Vcc), dihubungkan ke tegangan *power supply* DC +5 volt.

## 2.2 Pengenalan Bahasa C

### 2.2.1 Bahasa C untuk MCS51

Bahasa C tergolong ke dalam bahasa pemrograman tingkat menengah hal ini bukan berarti bahasa C kurang ampuh atau lebih sulit dibandingkan dengan bahasa pemrograman tingkat tinggi seperti Pascal dan BASIC melainkan menegaskan bahwa C bukanlah bahasa yang berorientasi pada bahasa mesin yang merupakan ciri bahasa pemrograman tingkat rendah. Pada kenyataannya C mengkombinasikan elemen dalam bahasa beraras tinggi dan bahasa beraras rendah. Bahasa C memiliki kelebihan diantaranya :kode bahasa C sifatnya portabel dan fleksibel untuk semua jenis komputer, bahasa C hanya menyediakan sedikit kata-kata kunci, proses executable program bahasa C lebih cepat serta dukungan pustakanya banyak. Bahasa C adalah bahasa yang terstruktur

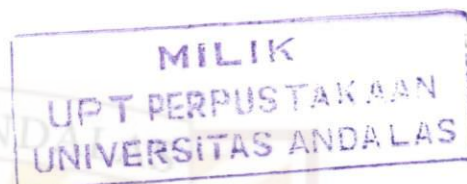
### 2.2.2 Bentuk umum program C

Pada dasarnya program C terdiri atas

```
#include <header>
```

```
Void main(void)
```

```
{
```





Deklarasi variabel;  
Deklarasi konstanta;  
Statemen;  
}

### 2.3 Motor DC

Motor DC merupakan motor listrik arus searah yaitu sebuah mesin yang dapat mengubah energi listrik menjadi energi mekanik. Motor yang mengoperasikan rancangan-rancangan kecil bekerja pada tegangan rendah yaitu 6 sampai 9 volt.

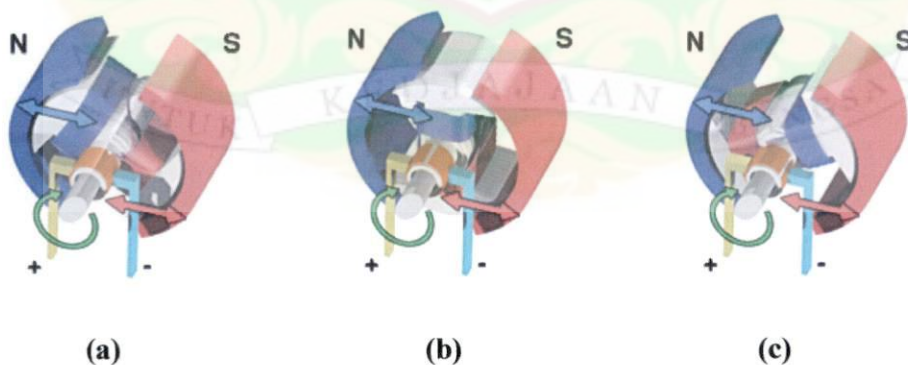
Motor arus searah memiliki magnet permanen yang memberikan medan magnet yang tetap. *Armatur* terdiri dari beberapa kumparan yang dililitkan pada inti besi dan dirangkaikan pada sebuah *komutator*, yang terdiri dari sepasang lempengan logam berbentuk setengah cincin. Cincin inilah yang terhubung dengan sumber tegangan. Bentuk fisik motor DC dapat dilihat pada Gambar 2.3.



**Gambar 2.3** Motor DC

### Prinsip kerja motor DC:

Pada Gambar 2.4 tampak bahwa ketika kumparan dialiri arus listrik, muncul medan magnet di sekitar *armatur*. Mari kita andaikan *armatur* yang menjadi kutub utara berwarna biru, dan yang selatan berwarna merah, seperti warna dari magnet pada sistem. *Armatur* biru akan ditolak oleh magnet bagian kiri karena kutubnya sama. Begitu pula dengan *armatur* merah, juga akan mengalami gaya tolak ke kanan. Akibatnya, sistem *armatur* berputar searah jarum jam. Saat *armatur* biru mendekati magnet kutub selatan, *armatur* biru mengalami gaya tarik ke arah sumbu horizontal dari kedua magnet. Saat posisi *armatur* horizontal, arus terputus sesaat karena penghubung tegangan positif dan negatif tepat berada pada celah cincin. Karena *armatur* masih berputar maka celah cincin akan terlewati dan cincin kembali mendapatkan tegangan listrik, namun dengan arah yang berlawanan. Karenanya dapat kita anggap bahwa *armatur* biru berubah warna menjadi merah, yang berarti proses berulang kembali. Demikian selanjutnya hingga motor itu terus berputar searah jarum jam selama masih diberi tegangan listrik. Hal ini ditunjukkan oleh Gambar 2.4.



**Gambar 2.4** Skema prinsip kerja motor DC

- (a) Terjadinya tolak-menolak antara kutub yang sama.
- (b) Munculnya gaya tarik-menarik antara *armatur* dengan kutub di seberangnya.
- (c) *Armatur* menuju posisi horizontal.

Pergerakan *armatur* memenuhi aturan Kaedah Tangan Kiri, yaitu: jika jari telunjuk searah dengan arah medan magnet, jari tengah menunjukkan arah arus maka ibu jari menunjukkan arah gaya Lorentz.

Besarnya gaya Lorentz yang dihasilkan adalah:

$$F = BIL \quad (2.1)$$

Keterangan:  $F$  = gaya Lorentz (N).

$B$  = fluks magnet (Weber).

$I$  = arus yang mengalir pada kumparan (A).

$L$  = panjang kumparan (m).

#### 2.4. Transistor sebagai Penguat Arus

Dari hukum *Kirchoff* diketahui bahwa jumlah arus yang masuk ke satu titik akan sama jumlahnya dengan arus yang keluar. Jika teorema tersebut diaplikasikan pada transistor, maka hukum itu menjelaskan hubungan :

$$I_E = I_C + I_B \quad (2.2)$$

Persamaan (2.2) tersebut mengatakan arus emiter  $I_E$  adalah jumlah dari arus kolektor  $I_C$  dengan arus basis  $I_B$ . Karena arus  $I_B$  sangat kecil sekali atau disebutkan  $I_B \ll I_C$ , maka dapat dinyatakan :

$$I_E = I_C \quad (2.3)$$

Beta didefinisikan sebagai besar perbandingan arus kolektor dengan arus basis.

$$\beta = \frac{I_C}{I_B} \quad (2.4)$$



Keterangan:

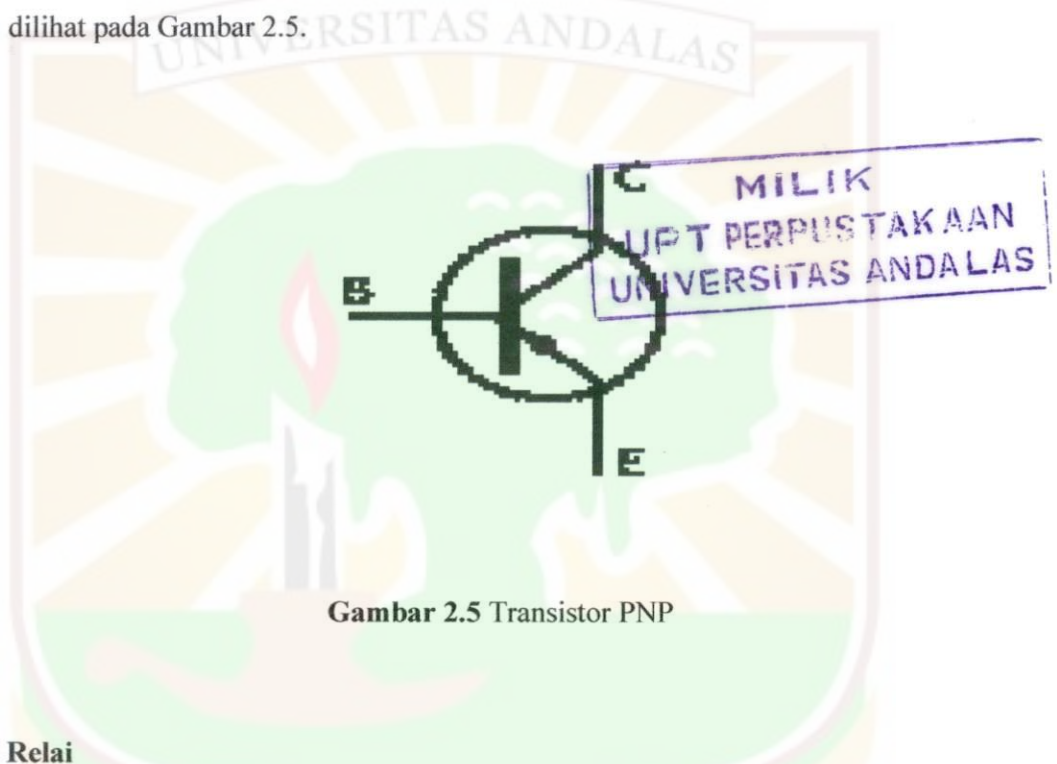
$\beta$  = penguatan arus.

$I_C$  = arus pada kolektor (A).

$I_B$  = arus pada basis (A).

$I_E$  = arus pada emitor (A) .

Dengan kata lain,  $\beta$  adalah parameter yang menunjukkan kemampuan penguatan arus (*current gain*) dari suatu transistor. Simbol transistor PNP dapat dilihat pada Gambar 2.5.

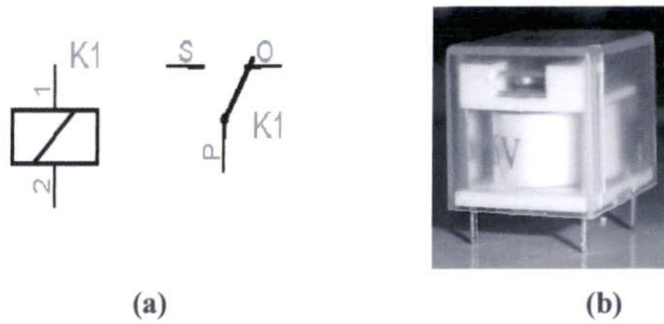


Gambar 2.5 Transistor PNP

## 2.5 Relai

Relai merupakan sebuah saklar yang dikendalikan oleh arus. Relai memiliki sebuah kumparan tegangan yang dililitkan pada sebuah inti. Kontak-kontak relai akan tertarik menuju inti apabila arus mengalir menuju kumparan. Kontak-kontak dapat ditarik apabila garis magnet dapat mengalahkan gaya pegas yang melawannya. Simbol dan bentuk fisik relai dapat dilihat pada Gambar 2.6.





**Gambar 2.6** Relai (a) Simbol relai (b) Bentuk fisik relai

Kontak-kontak atau kutub-kutub dari relai umumnya memiliki tiga dasar pemakaian diantaranya : bila kumparan dialiri arus listrik maka kontakannya akan menutup dan disebut sebagai kontak *Normally Open* ( NO ), bila kumparan dialiri arus listrik maka kontakannya akan membuka dan disebut dengan kontak *Normally Close* ( NC ) serta tukar-sambung ( *Change Over/CO* ), relai jenis ini mempunyai kontak tengah yang normalnya tertutup tetapi melepaskan diri dari posisi ini dan membuat kontak dengan yang lain bila relai dialiri listrik.

## 2.6 Hubungan roda-roda

Sistem ban berjalan dirancang dengan menggunakan 2 roda yang dihubungkan dengan sabuk. Kemudian sistem ini dihubungkan dengan motor DC dan gerakan dari ban berjalan mengikuti arah dan kecepatan motor DC. Untuk gerak melingkar yang terjadi pada roda-roda yang saling berhubungan ada 3 macam :

### 2.6.1 Roda-roda sepusat

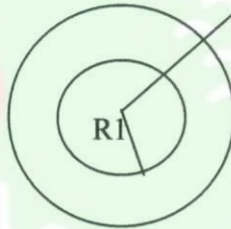
Pada roda-roda yang dihubungkan sepusat kecepatan sudut kedua roda adalah sama sedangkan kecepatan linier kedua roda adalah berbeda.

$$\omega_1 = \omega_2 \quad (2.5)$$

karena  $\omega = \frac{v}{R}$  maka

$$\frac{v_1}{R_1} = \frac{v_2}{R_2} \quad (2.6)$$

Skema hubungan roda-roda sepusat dapat dilihat pada Gambar 2.7.



**Gambar 2.7** Roda-roda sepusat

### 2.6.2 Roda-roda bersinggungan

Pada umumnya hubungan roda-roda yang bersinggungan ini bertujuan untuk mengubah arah gerak seperti terlihat pada Gambar 2.8.



**Gambar 2.8** Roda-roda yang bersinggungan

Pada hubungan roda–roda yang bersinggungan kecepatan linier kedua roda sama, tetapi arahnya berlawanan dan kecepatan sudutnya berbeda.

$$v_1 = v_2 = v$$

Karena  $v = \omega R$  maka diperoleh :

$$\omega_1 R_1 = \omega_2 R_2 \quad (2.7)$$

### 2.6.3 Roda–roda yang dihubungkan dengan sabuk

Pada roda–roda yang dihubungkan dengan sabuk pada prinsipnya sama dengan roda–roda yang saling bersinggungan, bedanya adalah roda yang dihubungkan dengan sabuk tidak mengubah arah gerak, arah putar kedua roda sama, kecepatan linier kedua roda sama tetapi kecepatan sudut kedua roda berbeda. Skema roda-roda yang dihubungkan dengan sabuk dapat dilihat pada Gambar 2.9.

$$v_1 = v_2 = v$$

$$\omega_1 R_1 = \omega_2 R_2$$



**Gambar 2.9** Roda–roda yang dihubungkan dengan sabuk

## 2.7 Perangkat Lunak Mikrokontroler AT89S51

Perangkat lunak adalah rangkaian instruksi atau perintah yang berfungsi untuk menjalankan perangkat keras dan mengolah data atau mewujudkan keinginan pemakai mikrokontroler mendapatkan informasi

dalam proses pengambilan keputusan. Tanpa perangkat lunak, mikrokontroler hanyalah barang rongsokan yang tidak dapat digunakan sama sekali. Perangkat lunak secara umum dapat dibedakan menjadi lima golongan, yaitu: sistem operasi, program bantu atau program utilitas, program paket, program aplikasi dan bahasa pemrograman.

Bahasa pemrograman merupakan sebuah perangkat lunak yang diciptakan secara khusus untuk membuat perangkat lunak lainnya. Bahasa pemrograman merupakan suatu kode-kode yang digunakan untuk memerintahkan mikrokontroler atau komputer melakukan sesuatu. Menurut I Made Juni dan Budi Raharjo (2006), berdasarkan kedekatannya dengan bahasa manusia, bahasa pemrograman dibedakan atas tiga tingkatan yaitu: bahasa pemrograman tingkat rendah, bahasa pemrograman tingkat menengah dan bahasa pemrograman tingkat tinggi.

Bahasa pemrograman tingkat rendah mempunyai kecepatan yang lebih tinggi dari bahasa pemrograman tingkat lainnya. Namun bahasa pemrograman tingkat rendah termasuk bahasa yang sulit digunakan sebab kode-kode yang digunakan dalam bentuk biner atau lebih menjurus ke bahasa mesin. Kode bahasa tingkat menengah lebih mudah dipahami, karena bahasa yang digunakan lebih dekat dengan bahasa yang biasa digunakan manusia. Walaupun demikian bahasa ini masih sukar dipahami karena adanya kerumitan dalam penggunaannya. Contohnya bahasa *Assembly* dan bahasa *C*. Bahasa pemrograman tingkat tinggi sudah mendekati bahasa manusia. Pada umumnya bahasa yang digunakan adalah bahasa Inggris, sehingga



menyebabkan berkurangnya kecepatan dalam memproses data digital. Contoh dari bahasa tingkat ini adalah bahasa pemrograman *Delphi* dan *Visual Basic*.

Sebuah mikrokontroler tidak akan bekerja apabila tidak diberikan program atau instruksi kepadanya. Pada dasarnya program menjalankan instruksi demi instruksi secara berurutan. Meskipun demikian, program yang kerjanya hanya berurutan saja tidaklah begitu berarti, untuk keperluan ini mikrokontroler dilengkapi dengan instruksi-instruksi untuk mengatur alur program. Program C merupakan program yang ditulis oleh pembuat program berupa kumpulan baris-baris perintah dan biasanya disimpan dengan *extension* C ataupun *Editor DOS*. Bahasa C yang digunakan akan *dicompile* dengan *SDCC*, yaitu *compiler* bahasa C untuk berbagai mikrokontroler termasuk *MCS-51*. Program tersebut memberitahukan jenis mikrokontroler apa yang akan dikerjakan. Pada dasarnya program C terdiri atas :

```
#include <header>
```

```
Void main (void)
```

```
{
```

```
Deklarasi variable;
```

```
Deklarasi konstanta;
```

```
Statement;
```

```
}
```

Komentar dapat diberikan dengan menggunakan tanda // atau /\*.....\*/ , dengan catatan tanda // hanya dapat digunakan untuk komentar 1 baris. Program C minimal harus memiliki satu fungsi yaitu fungsi *main*. *Void* di depan *main* menandakan bahwa fungsi *main* tidak memiliki nilai balik.

## 2.8 Pewaktuan Mikrokontroler AT89S51

Pewaktuan dapat dilihat dari frekuensi kristal yang digunakan yaitu dengan menggunakan persamaan :

$$T = 1/f \quad (2.8)$$

Keterangan:  $T$  = perioda (s).

$F$  = frekuensi (Hz).

Jika kristal yang digunakan memiliki frekuensi 11.0592 MHz, maka perioda osilator yang dihasilkan adalah :

$$T = \frac{1}{f} = \frac{1}{11.0592 \text{ MHz}}$$

$$T = 0.09 \mu s$$

Satu siklus mesin terdiri atas 6 runtun keadaan (*state*), setiap keadaan terdiri atas dua periode osilator. Sehingga setiap siklus mesin terdiri atas 12 periode osilator, maka satu siklus mesin dari frekuensi osilator di atas yaitu:

$$T = 12 \times 0.09 \mu s$$

$$T = 1.08 \mu s$$

Nilai dari perioda satu siklus mesin ini dikali dengan angka-angka yang dimasukkan pada program.

## BAB III

### METODE PENELITIAN

Pada bab ini akan dibahas mengenai berbagai alat dan bahan serta rancangan rangkaian yang diperlukan dalam penelitian. Rancang bangun alat kontrol ban bejalan berbasis mikrokontroler AT89S51 ini meliputi rancang bangun perangkat keras dan rancang bangun perangkat lunak. Rancang bangun perangkat keras meliputi: rancangan rangkaian catu daya, rancangan rangkaian mikrokontroler, rancangan rangkaian driver, dan rancangan rangkaian pengatur kecepatan. Sedangkan rancang bangun perangkat lunak meliputi rancangan diagram alir dan pemograman alat dengan menggunakan instruksi-instruksi dalam bahasa C.

#### 3.1 Alat Yang Digunakan Dalam Penelitian

Alat-alat yang digunakan dalam penelitian ini antara lain: solder dan pencabut timah adalah alat untuk mematri komponen pada PCB, slide transparan sebagai lembaran untuk memindahkan rancangan rangkaian ke PCB, setrika untuk memanaskan slide transparan agar gambar rangkaian pada slide menempel pada PCB, timah adalah bahan untuk mematri, tang penjepit untuk mengencangkan baut, sedangkan obeng, gunting berfungsi untuk memasang dan membuka baut serta sebagai alat pemotong. Papan stripboard berfungsi sebagai tempat uji coba rangkaian,  $\text{FeCl}_3$  sebagai pelarut lapisan tembaga yang tidak diperlukan., PCB cetak sebagai tempat merangkai komponen rancang bangun, *Crocodile clip* sebagai penghubung, multimeter digital untuk menguji



rangkaian serta komponen-komponennya selanjutnya PC (*Personal Computer*) digunakan untuk merancang pemograman pada mikrokontroler

### **3.2 Tata Laksana Penelitian.**

Cara melaksanakan penelitian adalah sebagai berikut : pada awalnya dilakukan pengumpulan bahan penelitian melalui tinjauan pustaka dan internet, pengenalan karakteristik, fungsi dan penggunaan masing-masing komponen di dalam rangkaian. Kemudian dilakukan tahap uji coba yaitu, merangkai komponen di papan stripboard dan pengujian keluaran. Setelah perancangan tata letak komponen pada PCB cetak, komponen dirangkai kemudian disolder. Selanjutnya dilakukan uji keluaran per blok rangkaian, uji keluaran rangkaian secara keseluruhan. Kemudian dilanjutkan dengan pembuatan perangkat lunak dan diakhiri dengan pengujian akhir meliputi pengujian perangkat-keras dan perangkat-lunak.

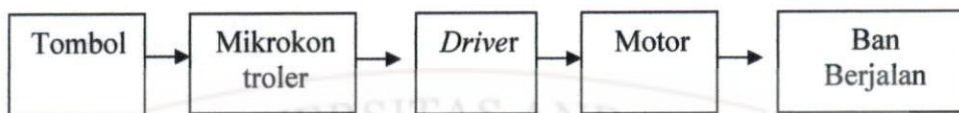
### **3.3 Rancang Bangun Alat Pengontrol Ban Berjalan Berbasis Mikrokontroler AT 89S51**

Rancang bangun Alat Pengontrol Sistem Ban Berjalan berbasis mikrokontroler AT89S51 dengan menggunakan motor DC secara umum dibagi menjadi dua bagian utama yaitu: rancang-bangun perangkat lunak dan rancang-bangun perangkat keras. Rancang bangun perangkat lunak berkaitan dengan program yang digunakan untuk menggerakkan motor DC, yang akan diproses mikrokontroler untuk melakukan putar kiri, putar kanan, dipercepat ataupun diperlambat kecepatannya. Sedangkan rancang bangun perangkat keras



berkaitan dengan rangkaian elektronika yang digunakan untuk mikrokontroler dan rangkaian *driver* yang berguna untuk menggerakkan motor DC

Diagram blok rancang bangun alat pengontrol sistem ban berjalan berbasis mikrokontroler AT89S51 dengan menggunakan motor DC dapat dilihat pada Gambar 3.1.

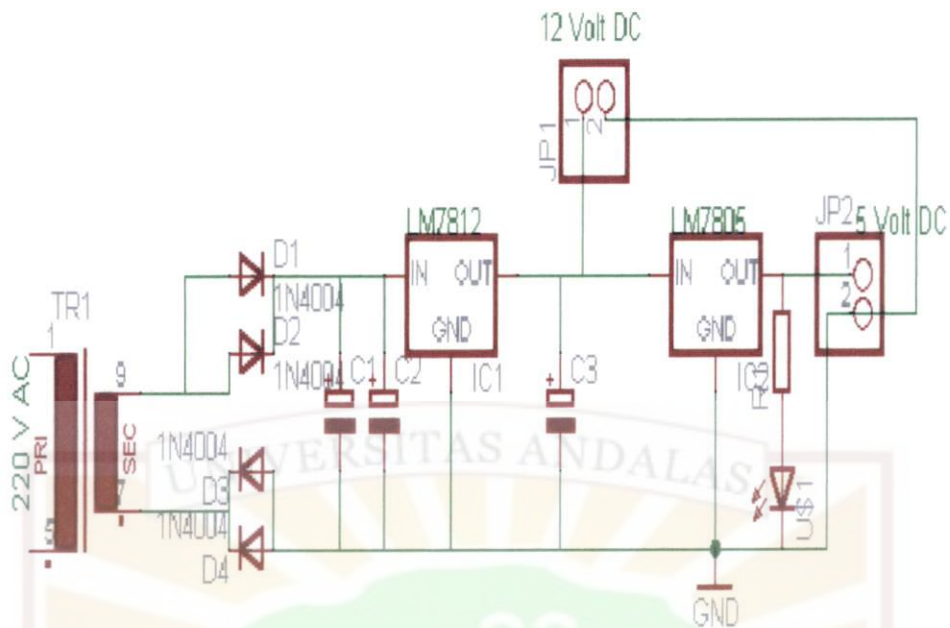


**Gambar 3.1** Diagram Blok Sistem Kontrol Alat

Jika tombol ditekan akan memberi perintah masukan kepada mikrokontroler, instruksi-instruksi tersebut digunakan untuk mengontrol kerja rangkaian *driver*. Rangkaian *driver* merupakan rangkaian untuk mengatur arah dan kecepatan motor DC. Gerakan motor DC ini dimanfaatkan untuk menggerakkan ban berjalan.

#### 3.4. Rangkaian Catu Daya

Catu daya berfungsi sebagai sumber arus DC yang dimanfaatkan sebagai tegangan masukan pada rangkaian mikrokontroler AT89S51 dan rangkaian *driver*. Diagram rangkaian Skematik catu daya 12 volt dapat dilihat pada Gambar 3.2.



**Gambar 3.2** Rangkaian catu daya 12 V

Tahap-tahap dalam pembuatan rangkaian catu daya 12 V sebagai berikut: komponen-komponen dirangkai pada PCB cetak seperti pada Gambar 3.2. Bahan yang dibutuhkan diantaranya : transformator step down 220 V 0.5 A sebanyak 1 buah berfungsi untuk menurunkan tegangan AC, dioda tipe 1N4004 1A sebanyak 4 buah berfungsi sebagai penyearah tegangan AC menjadi DC, kapasitor 1000  $\mu\text{F}$  25 V sebanyak 1 buah dan kapasitor 470  $\mu\text{F}$  25 V sebanyak 1 buah berfungsi sebagai filter, resistor 1K $\Omega$  sebanyak 1 buah berfungsi untuk memperkecil tegangan sebelum tegangan melewati LED, LED 5 mm sebanyak 1 buah berfungsi sebagai indikator, Pinhead 2 pin sebanyak 1 buah, IC LM7805 sebanyak 1 buah berfungsi untuk menghasilkan tegangan +5 V DC dan IC LM 7812 sebanyak 1 buah, berfungsi untuk menghasilkan tegangan +12 V DC. Rangkaian disolder kemudian diuji keluaran rangkaian secara keseluruhan.

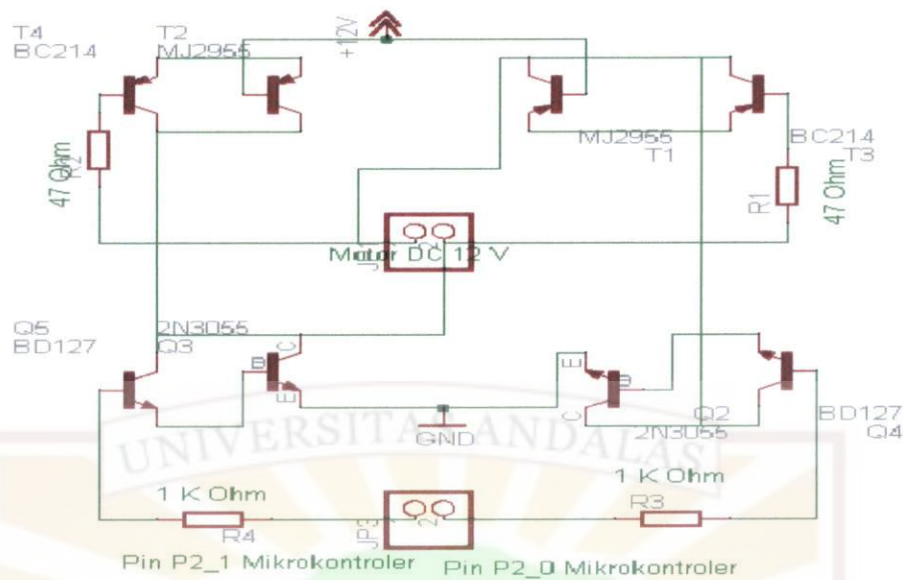




sebagai pusat pengolah dan pengendali data, kapasitor 30 pF dua buah berfungsi untuk menstabilkan kristal, kristal 11,0592 MHz satu buah berfungsi dalam pewaktuan, kapasitor 10  $\mu$ F satu buah berfungsi sebagai kapasitor *reset*, tombol *reset* 1 buah berfungsi untuk *merefresh*, resistor 1 k $\Omega$  satu buah berfungsi sebagai hambatan pada LED indikator, resistor 10 k $\Omega$  satu buah berfungsi untuk tombol *reset*, resistor 330  $\Omega$  tiga buah, berfungsi sebagai hambatan pada konektor penanaman, pinhead 5 pin satu buah berfungsi sebagai konektor penanam ke DB 25 PC, VCC +5 volt 1 buah dan GND. Setelah komponen-komponen tersebut dirangkai kemudian diuji tegangan keluarannya.

### 3.6 Rangkaian *driver*

Rangkaian *driver* merupakan rangkaian yang berfungsi untuk mengatur arah putaran motor DC. Proses pengendalian arah putaran dan kecepatan motor DC ini ditentukan oleh mikrokontroler AT89S51. Rangkaian dirancang pada dasarnya dengan menggunakan 4 buah transistor. Namun agar penguatan dari transistor lebih besar maka setiap transistor dibentuk dengan konfigurasi *darlington* dimana setiap transistor terdiri dari dua buah transistor, seperti yang terlihat pada Gambar 3.9.



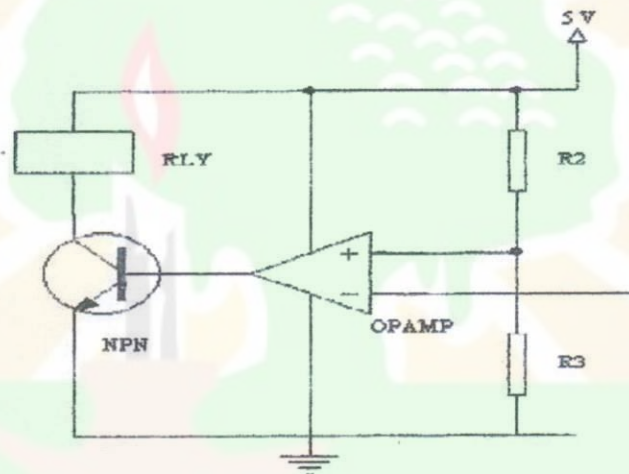
**Gambar 3.4 Rangkaian driver**

Berikut beberapa tahap dalam pembuatan rangkaian *driver*; pada awalnya komponen-komponen berikut ini dirangkai pada PCB cetak seperti pada Gambar 3.4. Bahan yang digunakan untuk rangkaian *driver* antara lain: transistor 2N3055 sebanyak 2 Buah, transistor C9013 sebanyak 2 buah, transistor MJ2955 sebanyak 2 buah. Transistor C9012 sebanyak 2 buah, resistor  $1k\Omega$  sebanyak 2 buah, resistor  $4,7 k\Omega$  sebanyak 2 buah, motor DC 1 buah dan pinhead.

Transistor-transistor tersebut berfungsi sebagai saklar dan sebagai penguat arus dan tegangan pada rangkain, sedangkan resistor berfungsi sebagai pembagi tegangan serta motor berfungsi untuk menggerakan ban berjalan. Setelah rangkaian disolder kemudian diuji keluaran rangkaian secara keseluruhan

### 3.7 Rangkaian Pengatur Kecepatan

Rangkaian pengatur kecepatan pada dasarnya merupakan rangkaian saklar elektronik dengan menggunakan transistor dan relai. Transistor dibuat bekerja pada dua keadaan yaitu keadaan *saturasi* dan *cut off*. Jika transistor dalam keadaan *saturasi*, arus akan mengalir maksimum pada kolektor dan relai terhubung. Jika transistor dalam keadaan *cut off*, maka arus minimum pada kolektor dan relai tidak terhubung. Skema rangkaian pengatur kecepatan dapat dilihat pada Gambar 3.5.



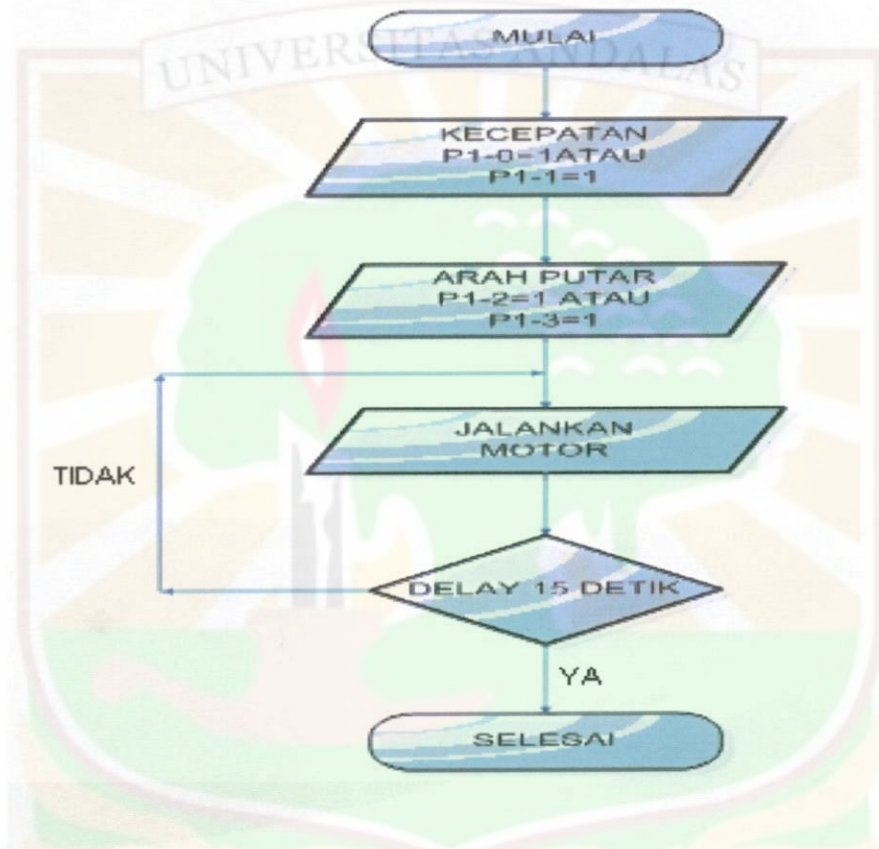
**Gambar 3.5** Relai

Bahan-bahan yang digunakan untuk rangkaian ini antara lain : resistor  $4,7\text{ k}\Omega$  dua buah berfungsi sebagai pembagi tegangan, relai satu buah berfungsi sebagai saklar dalam mengatur kecepatan serta IC 741 satu buah sebagai komparator. Komponen-komponen di atas dirangkai seperti Gambar 3.5 kemudian disolder dan selanjutnya diuji tegangan keluarannya. Penggabungan dari rangkaian-rangkaian diatas atau rangkaian lengkap dapat dilihat pada lampiran 4.



### 3.8 Rancangan Perangkat Lunak

Tahap awal dari perancangan program alat kontrol ban berjalan adalah membuat diagram alir program, sebelum menulisnya kedalam bahasa pemograman C. Ada pun diagram alir dari program tersebut dapat dilihat pada Gambar 3.6.



**Gambar 3.11** Diagram alir rancang bangun Alat Kontrol Ban Berjalan ber basis mikrokontroler AT89S51 dengan menggunakan Bahasa C

Dari diagram alir di atas dapat dijelaskan antara lain : mula-mula aktifkan mikrokontroler dengan mempersiapkan *interrupt* 0 . Setelah itu pilih tombol kecepatan yaitu, jika  $P1.0 = 1$  maka motor bergerak dengan kecepatan 1 atau jika  $P1.1=1$ , maka motor bergerak dengan kecepatan 2. Kemudian pilih arah putaran motor. Jika  $P1.0=1$  ,maka motor berputar ke kiri

dan jika  $P1.1=1$  maka motor akan berputar ke kanan. Selanjutnya jalankan motor sesuai perintah (kecepatan dan arah putar motor sesuai dengan pilihan di atas). Motor dijalankan selama 20 detik. Jika waktu belum mencapai 20 detik, motor akan terus berputar dan jika waktu sudah mencapai 20 detik motor akan berhenti akhirnya proses selesai.

Untuk perangkat lunak penulis menggunakan bahasa pemrograman C. Instruksi-instruksi yang digunakan dalam bahasa pemrograman C adalah untuk mengontrol kerja alat pengontrol sistem ban berjalan berbasis mikrokontroler AT98S51. Bahasa pemrograman untuk alat ini ini dapat dilihat pada lampiran 2. Bahasa pemrograman alat kontrol ban berjalan ini dibuat dengan menggunakan *software* M-IDE51 dengan tahap-tahap seperti yang terlihat pada lampiran 3.

## BAB IV

### HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai hasil penelitian meliputi hasil perangkat keras, hasil pengujian tegangan keluaran rangkaian tiap blok, hasil pengukuran kecepatan dan berat beban maksimum yang dapat dipindahkan oleh ban berjalan.

#### 4.1 Perangkat keras

Setelah melewati tahapan perancangan dan pembuatan serta *finishing* maka diperoleh sebuah perangkat keras rancang bangun sistem kontrol ban berjalan berbasis mikrokontroler AT89S51 dengan menggunakan motor DC dapat dilihat seperti pada Gambar 4.1



**Gambar 4.1** Perangkat keras rancang bangun sistem kontrol ban berjalan berbasis mikrokontroler AT89S51 dengan Menggunakan motor DC.



Hasil yang didapat dari pengukuran catu daya antara lain : tegangan masukan dari PLN sebesar +220 volt, tegangan keluaran IC LM 7805 sebesar +5 volt dan tegangan keluaran dari IC LM 7812 sebesar 12 volt.

Untuk menurunkan tegangan 220 volt AC (*Alternating Current*) menjadi tegangan 12 volt AC digunakan transformator step-down 0,5 ampere. Tegangan yang dikeluarkan oleh transformator tersebut masih berupa tegangan AC, untuk mengubahnya menjadi tegangan DC (*Direct Current*) digunakan rangkaian penyearah tegangan. Rangkaian penyearah tegangan tersebut terdiri atas 4 buah dioda yang saling berhubungan yang lebih dikenal sebagai rangkaian penyearah jembatan. Keluaran dari penyearah jembatan tersebut sudah berupa tegangan DC sebesar 12 volt yang akan digunakan sebagai masukan pada IC regulator LM7805 untuk menghasilkan tegangan keluaran DC sebesar 5 volt dan pada IC LM7812 menghasilkan tegangan 12 volt.

#### **4.2 Hasil Pengujian**

Pengujian perangkat keras alat kontrol ban berjalan berbasis mikrokontroler AT89S51 ini meliputi pengujian masing-masing blok rangkaian dan pengujian rangkaian secara keseluruhan. Hal ini dimaksudkan untuk mengetahui apakah sistem yang telah dirancang bangun tersebut dapat bekerja dengan baik. Hasil yang diperoleh dari pengujian rangkaian catu daya berupa tegangan masukan sebesar +220 volt, tegangan keluaran dari IC LM 7805 adalah +5 volt dan tegangan keluaran dari IC LM 7812 adalah sebesar + 12 volt.

Pada rangkaian minimum tegangan catu ( $V_{cc}$ ) yang masuk sebesar + 5 volt sedangkan tegangan yang diperlukan untuk relai sebesar +5 volt. Pada rangkaian *driver* tegangan catu ( $V_{cc}$ ) sebesar +12 volt dan tegangan masukan untuk memutar motor DC sebesar + 10,5 volt.

#### 4.3 Data hasil pengukuran kecepatan ban berjalan

Setelah melakukan pengujian terhadap kecepatan dari ban berjalan diperoleh data-data hasil pengukuran kecepatan ban berjalan meliputi data kecepatan 1 untuk putar ke kiri, data kecepatan 1 untuk putar ke kanan, data kecepatan 2 untuk putar ke kiri, dan data kecepatan 2 untuk putar ke kanan. Data hasil pengukuran untuk kecepatan 1 dengan arah gerakan ke kiri dapat dilihat pada Tabel 4.1.

**Tabel 4.1** Kecepatan 1 dengan arah gerak ke kiri

No	Kecepataan 1 arah ke kiri	
	Waktu untuk 10 putaran	Waktu untuk 1 putaran
1	10.42	1.042
2	10.50	1.050
3	10.44	1.044
4	10.59	1.059
5	10.58	1.058
6	10.44	1.044
7	10.36	1.036
8	10.41	1.041
9	10.36	1.036
10	10.27	1.027

Data hasil pengukuran kecepatan 1 ban berjalan dengan arah gerakan ke kanan dapat dilihat pada Tabel 4.2.

**Tabel 4.2** Kecepatan 1 dengan arah gerak ke kanan

No	Kecepataan 1 arah ke kanan	
	Waktu untuk 10 putaran	Waktu untuk 1 putaran
1	10,32	1,032
2	10,46	1,046
3	10,40	1,040
4	10,56	1,056
5	10,57	1,057
6	10,48	1,048
7	10,46	1,046
8	10,43	1,043
9	10,44	1,044
10	10,49	1,049

Data hasil pengukuran kecepatan 2 ban berjalan dengan arah gerakan ke kiri dapat dilihat pada Tabel 4.3.



**Tabel 4.3** Kecepatan 2 dengan arah gerak ke kiri

No	Kecepataan 2 arah ke kiri	
	Waktu untuk 10 putaran	Waktu untuk 1 putaran
1	6,75	0,675
2	6,53	0,653
3	6,58	0,658
4	7,25	0,725
5	6,99	0,699
6	6,59	0,659
7	6,61	0,661
8	6,51	0,651
9	6,53	0,653
10	6,59	0,659

Data hasil pengukuran kecepatan 2 ban berjalan yang bergerak dengan arah ke kanan dapat dilihat pada Tabel 4.4.

**Tabel 4.4** Kecepatan 2 dengan arah kanan.

No	Kecepataan 2 arah ke kanan	
	Waktu untuk 10 putaran	Waktu untuk 1 putaran
1	7,16	0,716
2	6,67	0,667
3	6,92	0,696
4	7,07	0,707
5	7,04	0,704
6	7,02	0,702
7	7,35	0,735
8	6,73	0,673
9	6,57	0,657
10	6,53	0,653

Dari keempat Tabel di atas yaitu tabel 4.1, Tabel 4.2, Tabel 4.3 dan Tabel 4.4 didapatkan nilai rata-rata kecepatan ban berjalan tanpa diberi beban seperti yang diperlihatkan oleh Tabel 4.5.

**Tabel 4.5** Kecepatan ban berjalan tanpa beban

No	Kecepatan	Arah Kecepatan	
		Kiri	Kanan
1	kecepatan 1	1,045 putaran/detik	1,046 putaran/detik
2	kecepatan 2	0, 6651 putaran /detik	0, 6906 putaran/detik

#### 4.4 Data hasil pengukuran beban maksimum

Berikut ini data hasil pengukuran beban maksimum yang dapat dipindahkan oleh ban berjalan dapat dilihat pada Tabel 4.6

**Tabel 4.6** Beban maksimum yang dapat dipindahkan oleh ban berjalan

No	Kecepatan	Beban Maksimum	
		Putar Kiri	Putaran Kanan
1	kecepatan 1	200 gram	220 gram
2	kecepatan 2	195 gram	190 gram

#### 4.5 Pembahasan

Hasil pengujian menunjukan alat berfungsi dengan baik. Tombol pertama merupakan *set mode* arah putar ke kiri sebagai masukan untuk P1.0. Tombol kedua merupakan *set mode* untuk putar kanan dan merupakan masukan untuk P1.1. Tombol ketiga merupakan *set mode* kecepatan 1 dan masukan bagi P1.2 dan tombol keempat merupakan *set mode* untuk kecepatan 2 dan masukan bagi P1.3 dimana kecepatan 1 lebih besar dari kecepatan 2. Jika tombol 3 dan 1 ditekan secara berurutan maka ban berjalan bergerak dengan kecepatan 1 dengan arah gerakannya ke kiri, jika tombol 4 dan tombol 2 ditekan secara berurutan maka ban berjalan bergerak dengan kecepatan 2 dan arah gerakannya ke kanan.



Motor DC yang penulis gunakan untuk menggerakkan ban berjalan adalah motor DC EG 500 AD-98 55203522. Pada Tabel 4.1 dapat dilihat bahwa besar kecepatan 1 untuk putar kiri 1,045 putaran per detik dan untuk arah kekanan 1,046 putaran perdetik. Untuk kecepatan 2 besarnya kecepatan ban berjalan ke arah kiri 0,665 putaran per detik dan kearah kanan 0,6906 putaran per detik. Selanjutnya beban maksimum yang dapat dipindahkan oleh ban berjalan dapat dilihat pada Tabel 4.2. Saat ban berjalan bergerak dengan kecepatan 1,045 ke arah kiri beban maksimum yang dapat dipindahkan adalah 200 gram, saat bergerak dengan kecepatan 1,046 ke kanan beban maksimum yang dapat dipindahkan adalah 220 gram, selanjutnya saat bergerak dengan kecepatan 0,6651 putaran per detik ke arah kiri beban maksimum yang dapat dipindahkan ban berjalan adalah 195 gram dan saat bergerak dengan kecepatan 0,6906 putaran per detik ke arah kanan beban maksimum yang dapat dipindahkan oleh ban berjalan adalah 220 gram. Dari data diatas terlihat bahwa besar kecepatan gerakan ban berjalan ke kiri dan ke kanan sedikit berbeda hal ini mungkin karena arus yang melewati rangkaian menuju ke kiri dan ke kanan motor tidak sama besar. Komponen-komponen yang digunakan misalnya resistor memiliki nilai toleransi ataupun kapasitor yang dipakai di sebelah kiri dan kanan motor memiliki kapasitas yang sedikit berbeda. Dari hasil pengujian beban maksimum yang dapat dipindahkan ban berjalan terlihat bahwa kemampuan alat ini sebanding dengan kecepatan, semakin besar kecepatan gerakan ban berjalan berat beban maksimum yang dapat dipindahkannya juga semakin besar. Efisiensi kerja alat kontrol ini sebesar 45,83 %.

## BAB V

### KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian yang telah dilakukan, penulis dapat mengambil kesimpulan sebagai berikut :

1. Alat pengontrol ban berjalan berbasis mikrokontroler AT89S51 dengan menggunakan motor DC dapat digunakan untuk memindahkan benda ataupun barang-barang dari suatu tempat ke tempat lain dengan beban maksimum 220 gram.
2. Alat pengontrol ban berjalan ini dapat bergerak dalam dua arah yang berlawanan (ke kiri dan ke kanan).
3. Alat pengontrol ban berjalan ini dapat diatur kecepatannya sesuai dengan kebutuhan.
4. Efisiensi kerja alat adalah sebesar 45,83 %.

Untuk penelitian selanjutnya penulis menyarankan sebagai berikut:

- 1 Agar ban berjalan ini dapat memindahkan barang dengan baik maka sebaiknya gunakan poros roda bergigi dan dihubungkan dengan rantai.
2. Agar alat dapat memindahkan beban yang lebih berat sebaiknya gunakan motor dengan daya yang lebih besar.

## DAFTAR PUSTAKA

- A, Grabel Higginbotham, D, 1981, Dasar-dasar Elektro Teknik, Erlangga, Jakarta.
- Andi Nalwan P, 2003, Paduan Praktis Teknik Antarmuka Pemrograman Mikrokontroler AT89C51, PT Elexmedia Komputindo Kelompok Gramedia, Jakarta
- Budioko, T, 2005, Belajar dengan Mudah dan Cepat Pemrograman Bahasa C dengan SDCC pada Mikrokontroler AT89S51/AT8951/52, Gava Media, Jogjakarta.
- Joni, I, M, Raharjo, B, 2006, Pemrograman C dan implementasinya, Informatika Bandung.
- Malvino, AP, Alih Bahasa Barmawi, Tjia, M, O, 1985, Prinsip-prinsip elektronika, Erlangga Jakarta.
- Mardiah, 2005, Rancang Bangun Sistem Kontrol Motor DC Berbasis Mikrokontroler AT89S51, Universitas Andalas, Padang.
- Simanjuntak, H, 2001, Dasar-dasar mikroprosesor, Penerbit Kasinius.
- Sutrisno, 1986, Elektronika teori dasar penerapan I, ITB, Bandung.
- Suryono dan Tim, 2005, Pemrograman dan Aplikasi Mikrokontroler ISP MCS 51, Workshop Fisika FMIPA, Universitas Diponegoro, Semarang
- Yandani, E, Belajar Mikrokontroler MCS51 menggunakan SDCC
- Data Sheet Mikrokontroler AP89S51.
- <http://www.electroniclab.com>, Agustus 2007
- <http://www.kelasmikrokontroler.com>
- [http://.widodo.com](http://widodo.com)





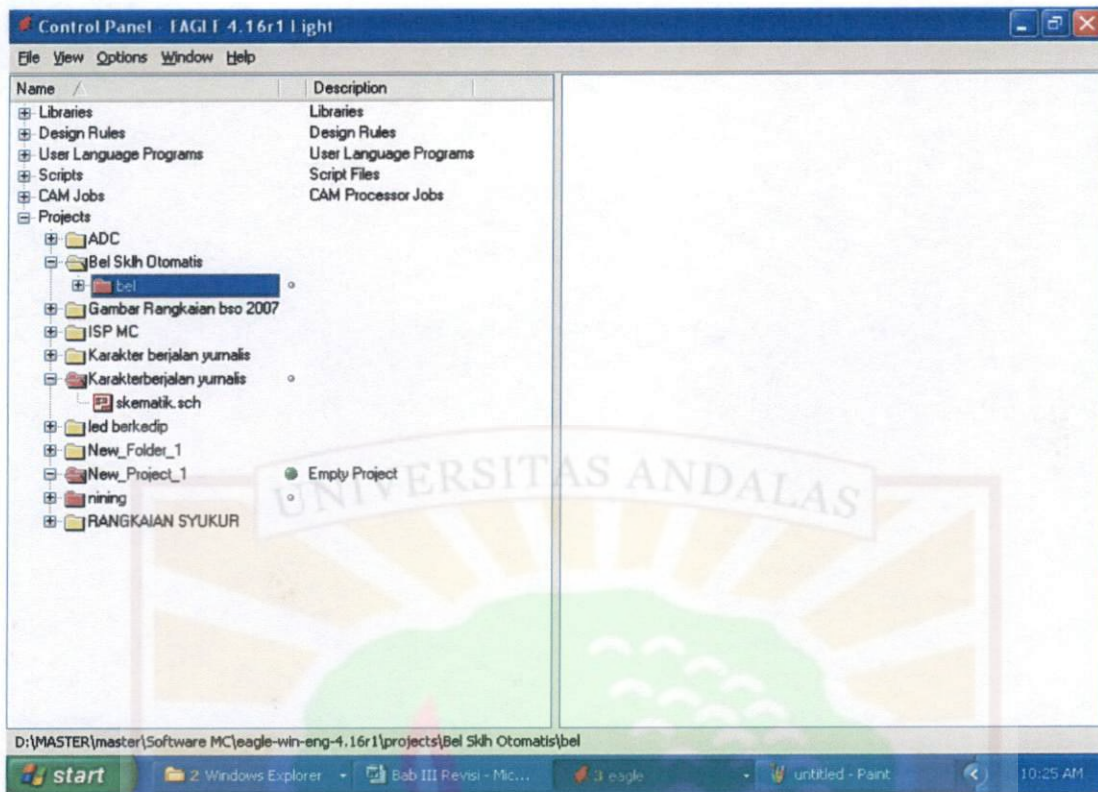
# *Lampiran*

**Lampiran 1.** Tahap-tahap pembuatan rangkaian dengan perangkat lunak eagle.

Berikut beberapa tahap dalam pembuatan rangkaian target mikrokontroler untuk alat pengontrol sistem ban berjalan.

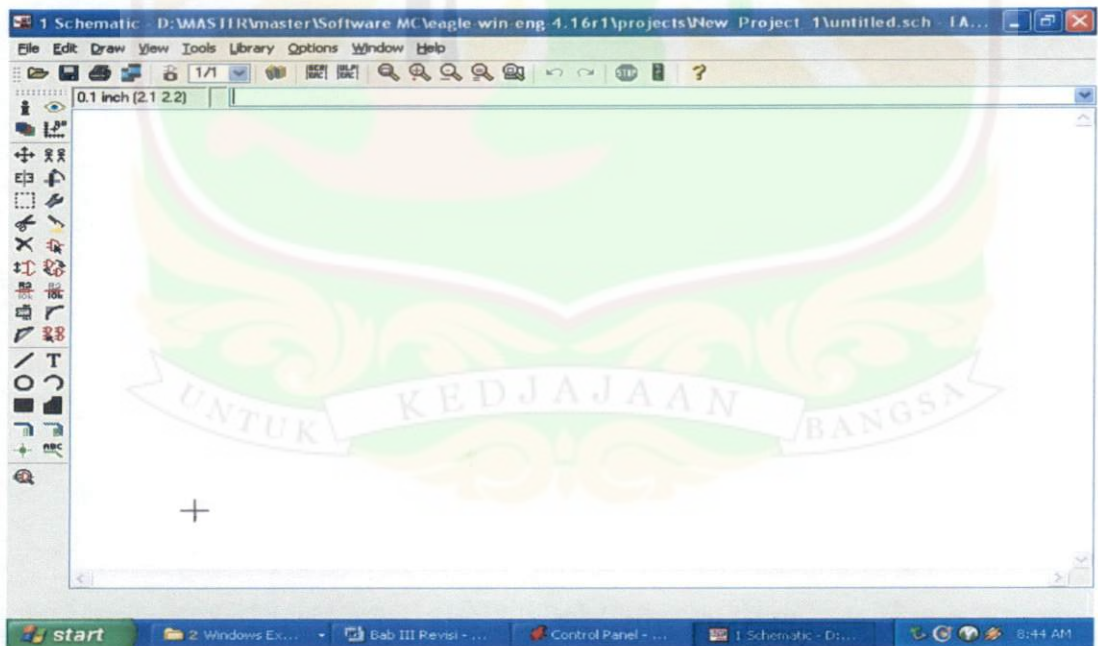
a. Pembuatan skematik (*Schematic*) dengan perangkat lunak *Eagle*.

- 1). Klik kiri dua kali *eagle-win-eng-4.16r1*.
- 2). Klik kiri dua kali bin.
- 3). *Delete eagle registration entries*.
- 4). Klik kiri dua kali *icon eagle* berwarna merah. Lalu muncul *control panel*.
- 5). Klik kanan *project*.
- 6). Klik kiri *new folder* dan beri nama ban berjalan
- 7). Klik kanan *folder* ban berjalan.
- 8). Klik kiri *new project* dan beri nama ban, dan muncul tampilan *control panel* dan dapat dilihat pada Gambar 1.



**Gambar 1.** *New project pada control panel.*

9). Klik kiri *new schematics*, muncul tampilan seperti Gambar 2,



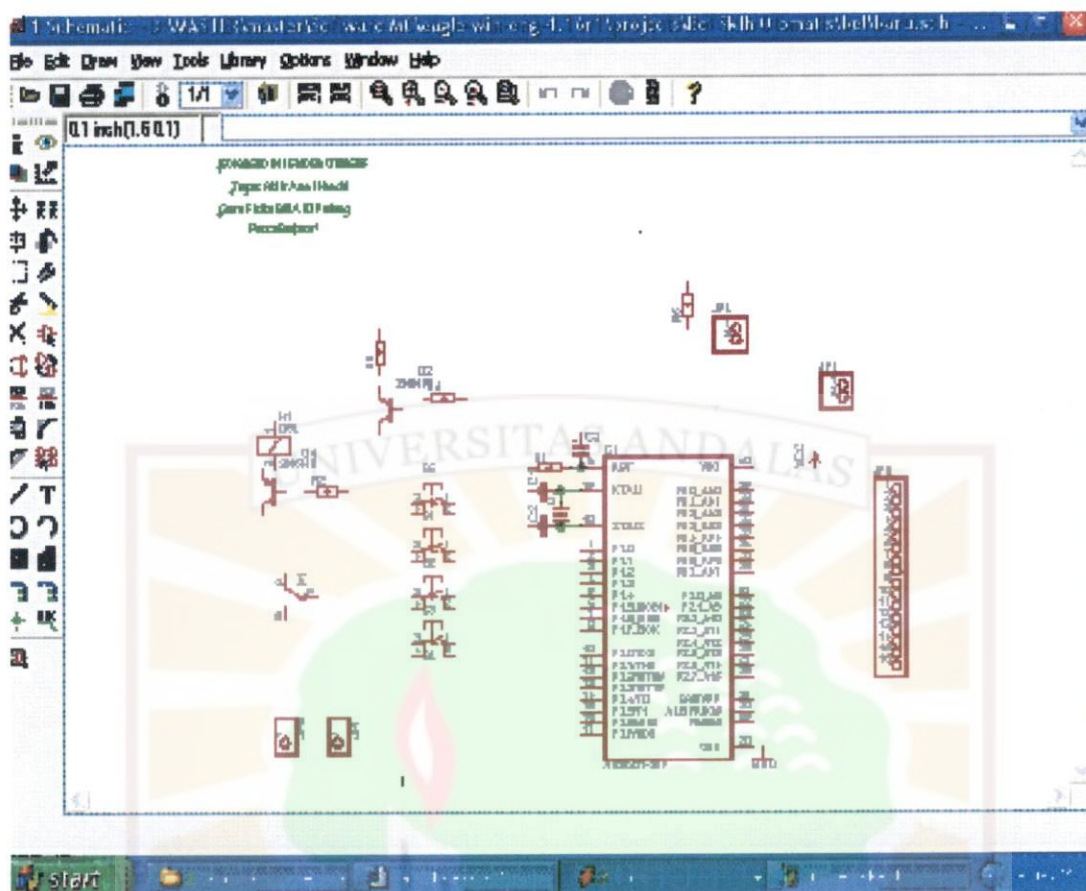
**Gambar 2.** *Stage schematic.*



10). Tambahkan komponen-komponen yang diperlukan untuk membuat rangkaian minimum seperti berikut ke *stage schematic*.

- Mikrokontroler AT89S51 satu buah, berfungsi sebagai pusat pengolahan dan pengendali data;
- Kapsitor 30 pF dua buah, berfungsi untuk menstabilkan kristal;
- Kristal 11,0592 MHz satu buah, berfungsi dalam pewaktuan;
- Kapasitor 10  $\mu$ F satu buah, berfungsi sebagai kapasitor *reset*;
- Tombol *reset* 1 buah, berfungsi untuk *refresh*;
- Resistor 1 k $\Omega$  satu buah, berfungsi sebagai hambatan pada LED indicator;
- Resistor 10 k $\Omega$  satu buah, berfungsi untuk tombol *reset*;
- Resistor 330  $\Omega$  tiga buah, berfungsi sebagai hambatan pada konektor penanaman;
- Pinhead 5 pin satu buah, berfungsi sebagai konektor penanam ke DB 25 PC;
- VCC +5 volt 1 buah;
- GND.

Penambahan komponen pada tampilan diatas menghasilkan tampilan seperti Gambar 3.

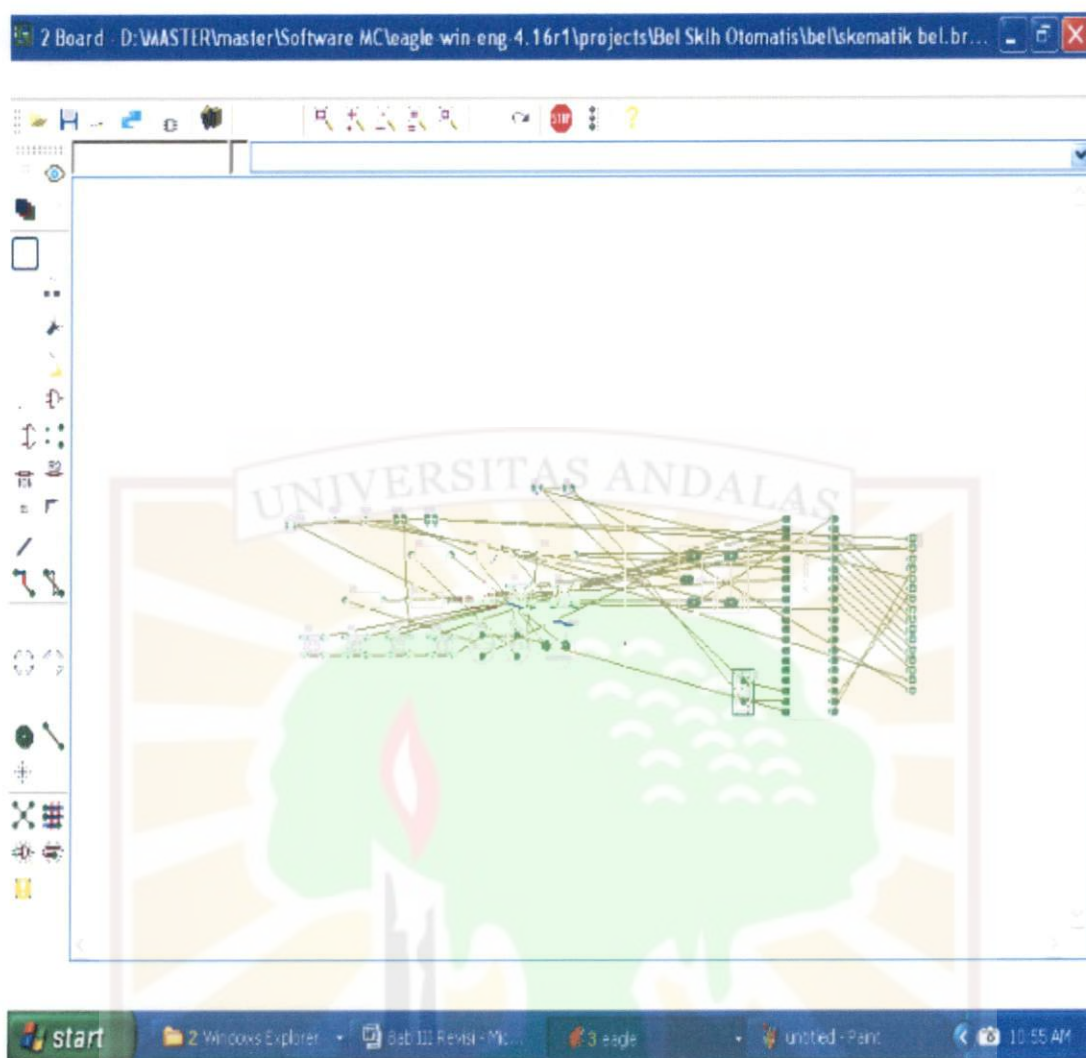


**Gambar 3.** Komponen yang telah ditambahkan ke *Stage schematic*.

- 11). Susun Komponen-komponen dengan mengaktifkan *move tool*.
- 12). Hubungkan komponen-komponen dengan mengaktifkan *wire tool*. *Schematic* ban berjalan terbentuk seperti pada Gambar .4

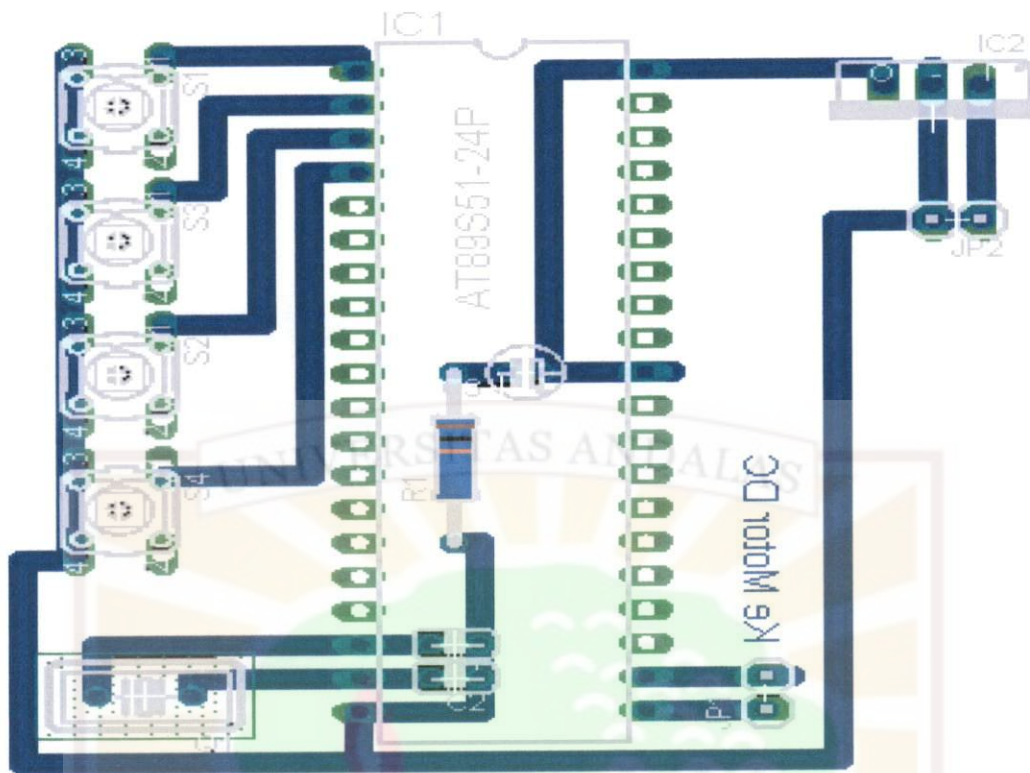






**Gambar 5.** Memindahkan komponen-komponen ke dalam persegi panjang.

- 4). Klik kiri DRC-klik kiri size-rubah nilai edit box 10 mil menjadi 30 mil pada label minimum width untuk memperbesar ukuran jalur ( *route* )-klik kiri check.
- 5). Klik kiri auto- rubah edit box | menjadi N/A pada general 1 top- klik ok.
- 6). Hasilnya seperti pada Gambar 6.



**Gambar 6.** Rangkaian ban berjalan dengan komponen.

- c. Mem-print dan meng-copy ke plastik transparan.
- d. Memindahkan ke PCB dengan sterika.
- e. Melubangi PCB.
- f. Membuang lapisan tembaga yang tidak diperlukan dengan  $\text{FeCl}_3$ .
- g. Menguji rangkaian dengan ohmmeter (multimeter).
- h. Memasang komponen ke *PCB*.
- i. Menyolder komponen ke *PCB*.
- j. Menguji lagi rangkaian dengan ohmmeter.

Dengan cara yang sama kita dapat membuat perangkat keras lainnya seperti catu daya, rangkaian *driver* dan rangkaian relai.

## Lampiran 2. Bahasa pemrograman C untuk mikrokontroler AT89S51.

```
#include "at89x51.h"
unsigned char d;
```

```
// =====
//   Ban berjalan           |
// ===== Juni 08 =====
```

```
void delay (unsigned int waktu)
{
    unsigned int i;
    for (i=0; i<waktu; i++);
}
void tunda (unsigned char waktu)
{
    unsigned int i;
    for (i=0; i<waktu; i++);
    {delay(1000);
    }
}
```

```
// Kecepatan dan arah putar
//-----//
```

Fungsi untuk putar kanan, P1.1 berlogika *higt*, dan pin yang lain berlogika *low*.

```
void putar_kanan()
{
    P1_0=0;
    P1_1=1;
    P1_2=0;
    P1_3=0;
    P1_4=0;
    P1_5=0;
    P1_6=0;
    P1_7=0;
    tunda (1000)
}
```

Fungsi untuk putar kiri, P1.0 berlogika *higt*, dan pin yang lain berlogika *low*.



```

void putar_kiri()
{
  P1_0=1;
  P1_1=0;
  P1_2=0;
  P1_3=0;
  P1_4=0;
  P1_5=0;
  P1_6=0;
  P1_7=0;
  tunda (1000)
}

```

Fungsi untuk kecepatan 1, P1.2 berlogika *hight*, dan pin yang lain berlogika *low*.

```

void speed_1()
{
  P1_0=0;
  P1_1=0;
  P1_2=1;
  P1_3=0;
  P1_4=0;
  P1_5=0;
  P1_6=0;
  P1_7=0;
  tunda (1000)
}

```

Fungsi untuk kecepatan 2, P1.3 berlogika *higt*, pin yang lain berlogika *low*.

```

void speed_2()
{
  P1_0=0;
  P1_1=0;
  P1_2=0;
  P1_3=1;
  P1_4=0;
  P1_5=0;
  P1_6=0;
  P1_7=0;
  tunda (1000)
}

```

```

void tombol()
{
  if(P1_0==0)

```

```

    {TR0=0;
    tunda(50)
    putar_kanan();
    TR0=1;}
if(P1_1==0)
    {TR0=0;
    tunda(50)
    putar_kiri();
    TR0=1;}
if(P1_2==0)
    {TR0=0;
    tunda(50)
    speed_1();
    TR0=1;}
if(P1_3==0)
    {TR0=0;
    tunda(50)
    speed_2();
    TR0=1;}
}

void interupsi(void) interrupt 1 using 1
{
    TH0 |=0xdc;
    d++;
}

void initjam()
{
    EA=1;
    ET0=1;
    TMOD |=0x01;
    TR0=1;
}

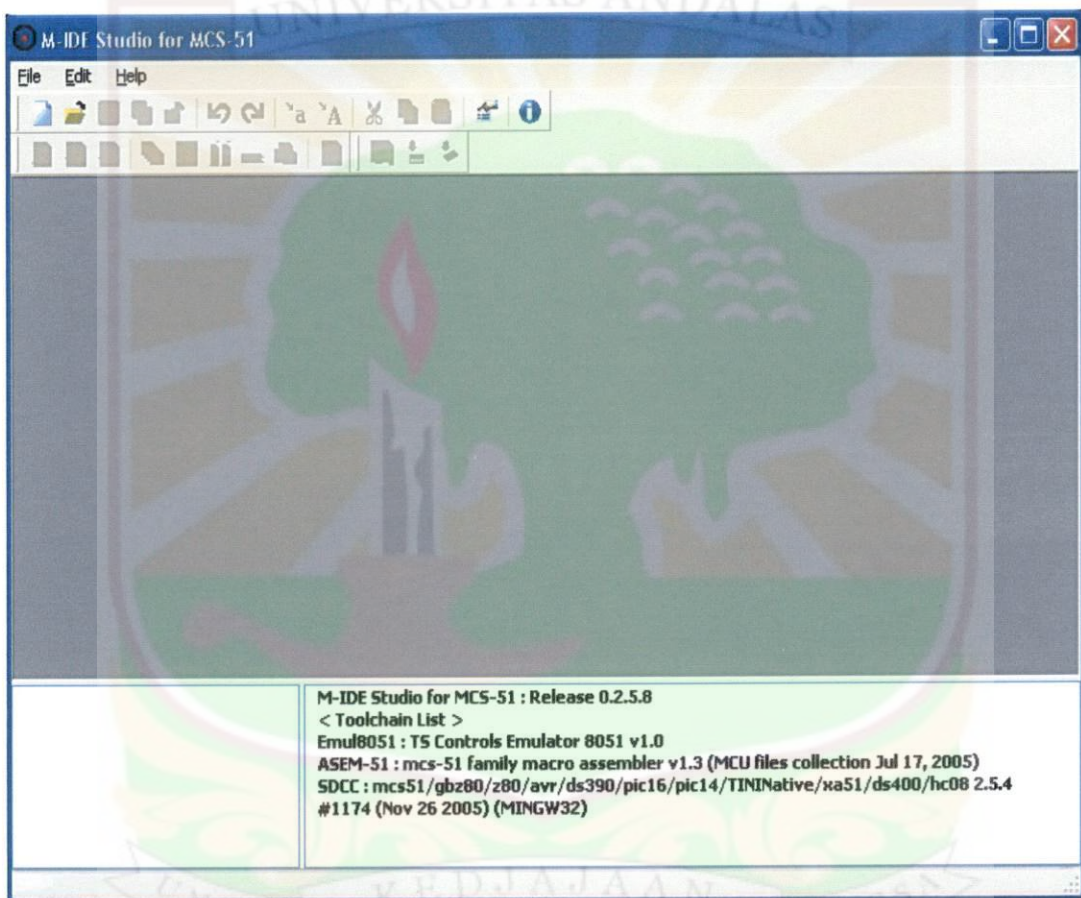
void main()
{
    initjam()
    while(1)
    { tombol();
    }
    tunda(20000)
    }

```

**Lampiran 3.** Tahap-tahap pemrograman mikrokontroler dengan menggunakan perangkat lunak M- IDE 51.

Program sistem kontrol ban berjalan di atas dibuat dengan menggunakan *software* M-IDE51 dengan tahap-tahap sebagai berikut :

1. Buka M-IDE51 sehingga muncul tampilan seperti Gambar .7.

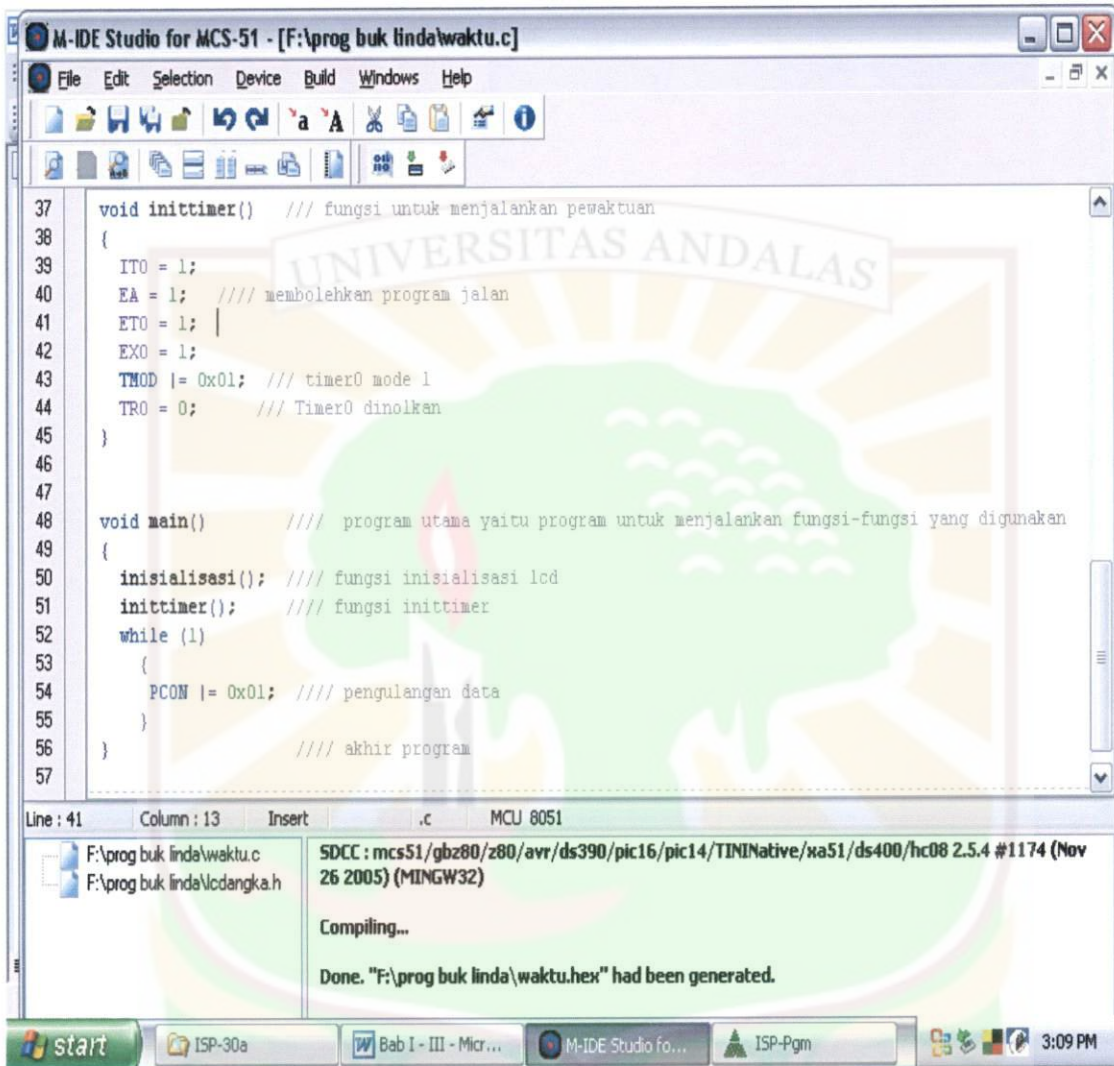


**Gambar 7.** Tampilan awal M-IDE51

2. Buatlah file C baru dengan mengklik menu *file* dan memilih *new*.
3. Ketikkan program C seperti yang terdapat pada bagian 3.1.3.
4. Buka *file*, kemudian pilih *save* untuk menyimpan program yang telah dibuat dengan tipe C ( contoh : D:\prog\progl.c ).



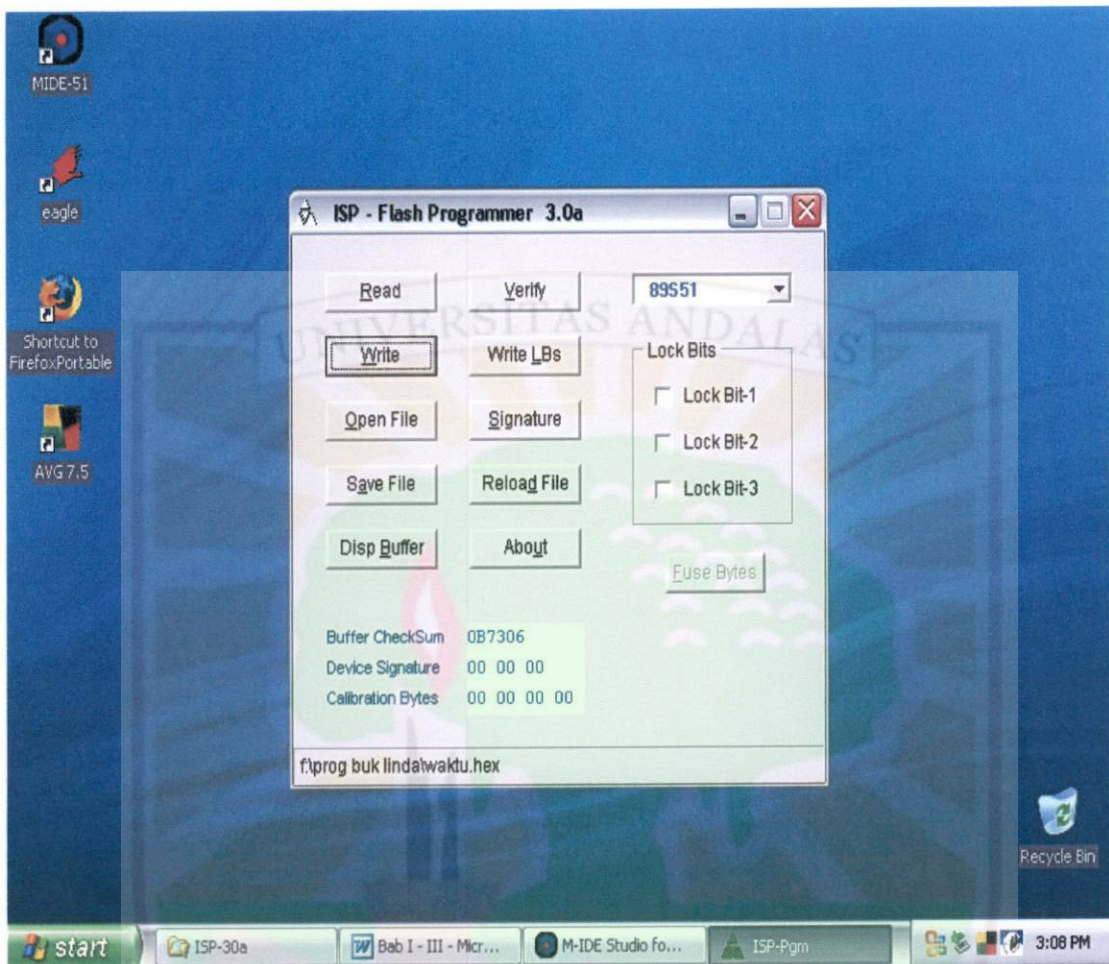
5. *Compile* program tersebut dengan mengklik tombol *build* pada *toolbar*, jika *compile*-an sukses maka akan dihasilkan *file* berekstensi **.hex** pada *folder* tempat kita menyimpan file C tadi seperti Gambar 8.



**Gambar 8.** Kotak diálog hasil *Compile*.

6. Pindahkan *file* berekstensi **.hex** hasil dari *Compile* ke mikrokontroler AT89S51 dengan menggunakan ISP-Flash Programmer 3.0a dengan cara sebagai berikut :
- Pasang AT89S51 pada rangkaian target .
  - Hubungkan rangkaian target dengan konektor DB 25 pada komputer.
  - Beri rangkaian target catu daya 5 volt DC.

- d. Buka ISP-Flash Programmer 3.0a pada komputer sehingga tampilan seperti Gambar 9.

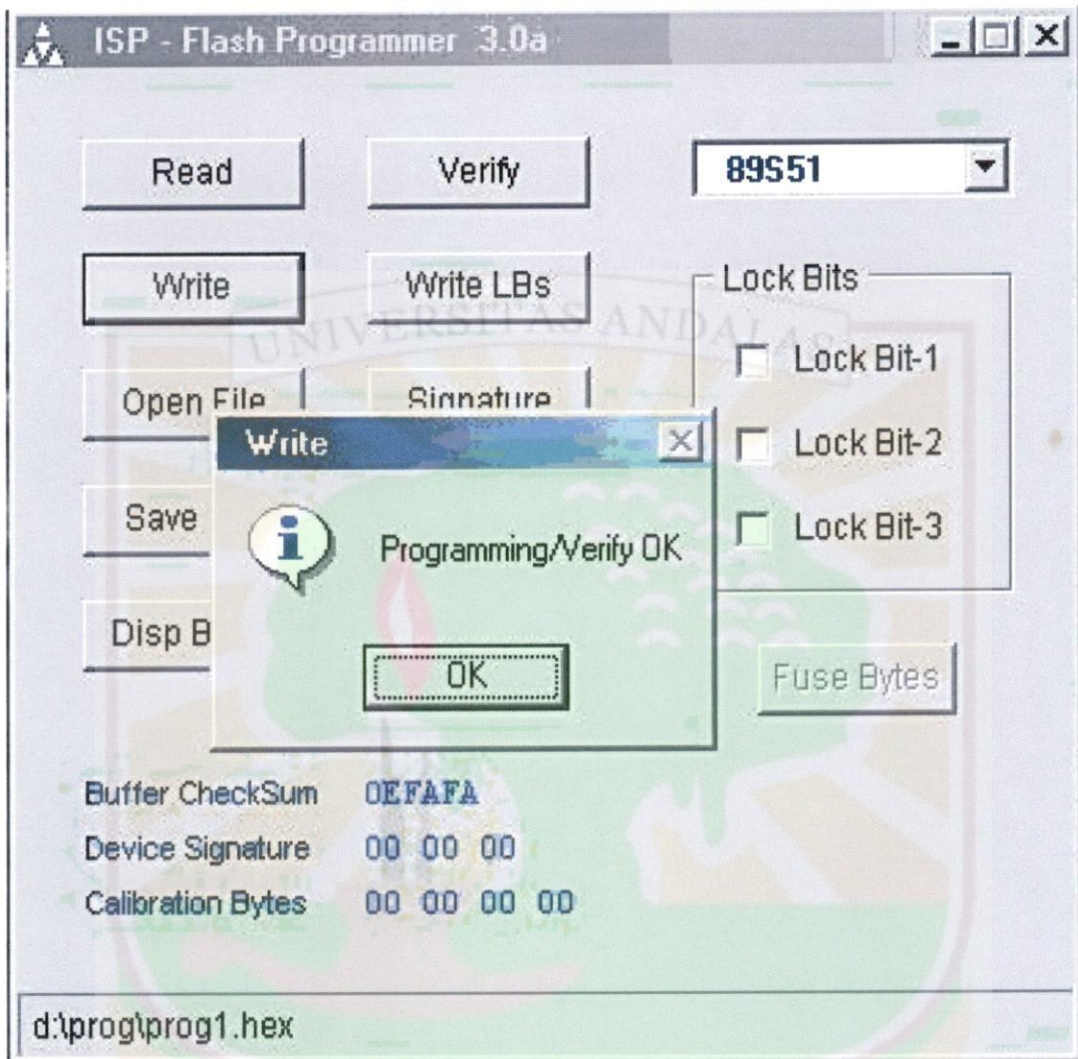


**Gambar 9.** Tampilan awal ISP-Flash Programmer 3.0a.

- e. Pilih AT89S51 untuk jenis mikrokontroler yang digunakan.
- f. Klik *Verify* untuk mengecek terhubung atau tidaknya komputer dengan rangkaian target.
- g. Jika tidak ada pesan *no response from  $\mu C$*  setelah *verify* (jika ada, periksa rangkaian target dan AT89S51-nya apakah sudah terpasang dengan benar), maka klik *Open File* untuk mengambil *file* berekstensi **.hex** pada *folder* tempat kita menyimpan *file C* tadi.



- h. Klik *Write* untuk menulis *file* berekstensi **.hex** tersebut ke rangkaian penanam mikrokontroler, sehingga tampil Gambar 10.

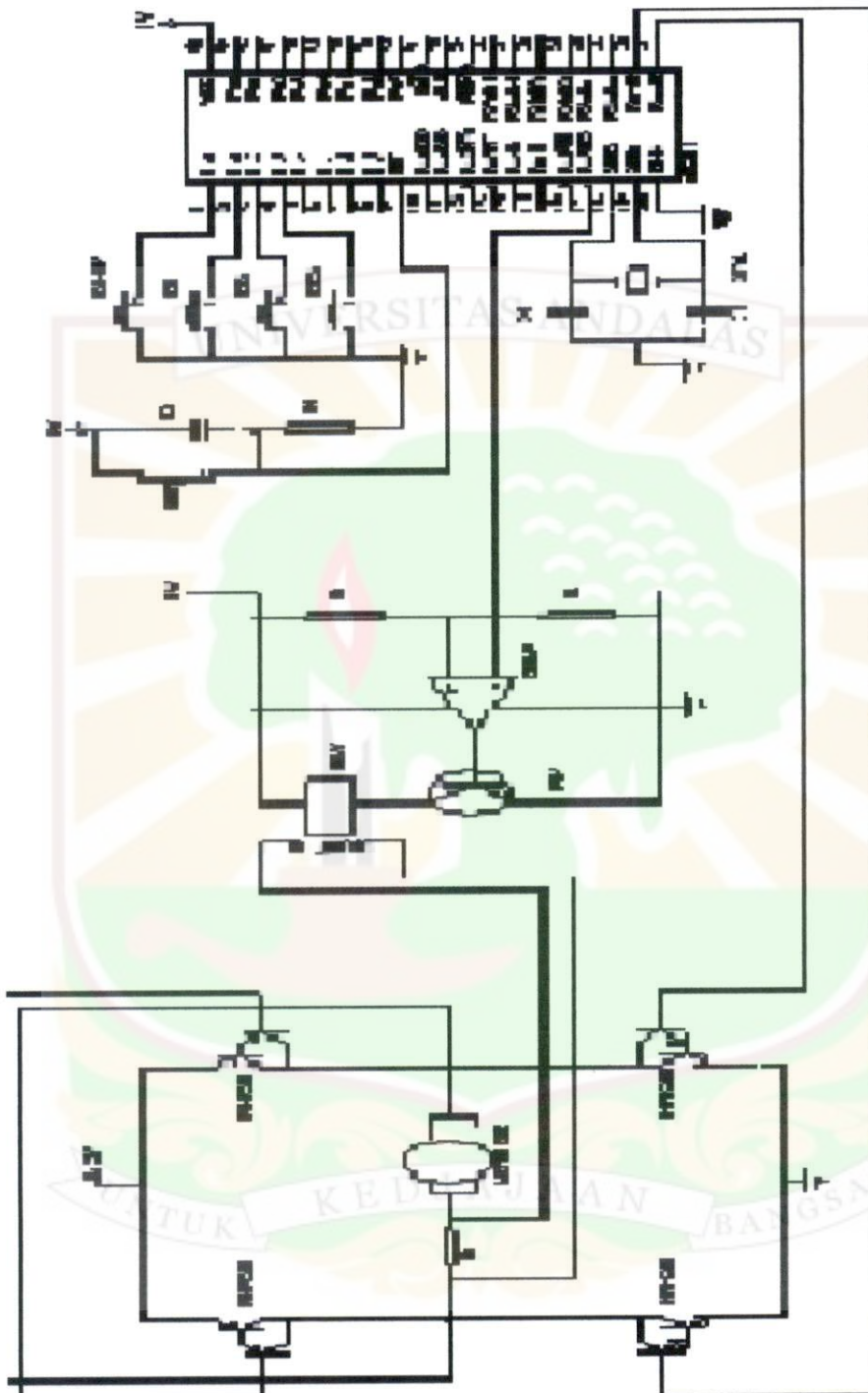


**Gambar 10.** Tampilan sukses pemindahan data ke mikrokontroler AT89S51.

- i. Terakhir pindahkan AT89S51 ke rangkaian mikrokontroler yang digunakan.
- Setelah penanaman program pada mikrokontroler AT89S51 selesai, berarti mikrokontroler sudah siap digunakan sebagai sistem Kontrol Ban Berjalan.



**Lampiran 4.** Rangkaian lengkap alat control ban berjalan berbasis mikrokontroler AT89S51.



## lampiran 5. Data sheet mikrokontroler AT89S51

### res

Compatible with MCS-51® Products  
 8K Bytes of In-System Programmable (ISP) Flash Memory  
 Endurance: 1000 Write/Erase Cycles  
 5.5V Operating Range  
 Static Operation: 0 Hz to 33 MHz  
 On-Chip Program Memory Lock  
 2K-bit Internal RAM  
 32 Programmable I/O Lines  
 16-bit Timer/Counters  
 5 Interrupt Sources  
 Full Duplex UART Serial Channel  
 Low Power Idle and Power-down Modes  
 Fast Recovery from Power-down Mode  
 Watchdog Timer  
 Data Pointer  
 Sleep Mode Flag  
 In-System Programming Time  
 In-System Programming (Byte and Page Mode)

### ption

The AT89S51 is a low-power, high-performance CMOS 8-bit microcontroller with 4K on-chip in-system programmable Flash memory. The device is manufactured using high-density nonvolatile memory technology and is compatible with the industry standard 80C51 instruction set and pinout. The on-chip Flash allows the program to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a single chip, the Atmel AT89S51 is a powerful microcontroller which provides a flexible and cost-effective solution to many embedded control applications.

The AT89S51 provides the following standard features: 4K bytes of Flash, 128 bytes of internal RAM, 32 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a five-level interrupt architecture, a full duplex serial port, on-chip oscillator, and low power consumption. In addition, the AT89S51 is designed with static logic for operation over a wide frequency range and supports two software selectable power saving modes. The Idle mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM content by freezing the oscillator, disabling all other chip functions until the next external hardware reset.



## 8-bit Microcontroller with 4K Bytes In-System Programmable Flash

### AT89S51

### Preliminary

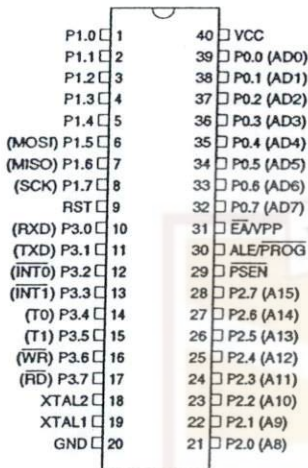
Rev. 2487A-10/01



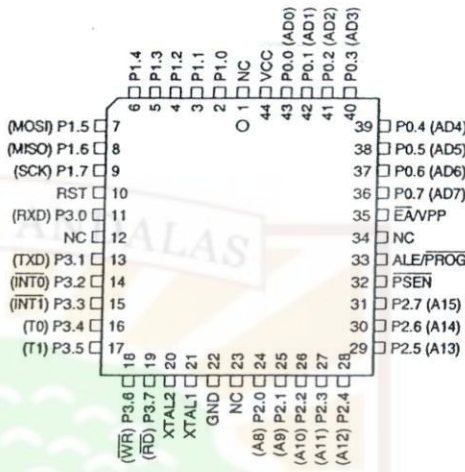


n Configurations

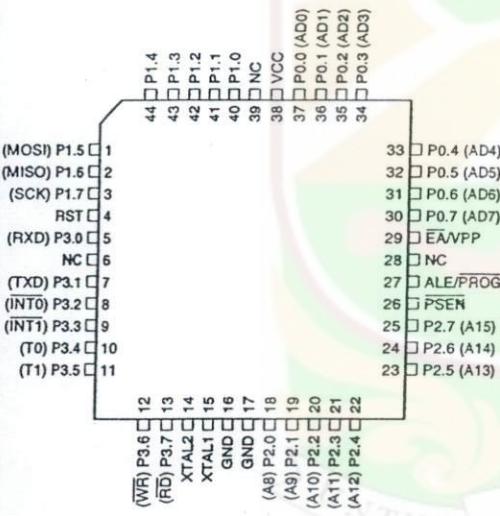
PDIP



PLCC



TQFP



MILIK  
UPT PERPUSTAKAAN  
UNIVERSITAS ANDALAS







n Description

CC

Supply voltage.

ND

Ground.

rt 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. **External pull-ups are required during program verification.**

rt 1

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the internal pull-ups.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port Pin	Alternate Functions
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

t 2

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the internal pull-ups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

3

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the pull-ups.

Port 3 receives some control signals for Flash programming and verification.

Port 3 also serves the functions of various special features of the AT89S51, as shown in the following table.



## AT89S51

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

Program Store Enable ( $\overline{\text{PSEN}}$ ) is the read strobe to external program memory.

When the AT89S51 is executing code from external program memory,  $\overline{\text{PSEN}}$  is activated twice each machine cycle, except that two  $\overline{\text{PSEN}}$  activations are skipped during each access to external data memory.

External Access Enable.  $\overline{\text{EA}}$  must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed,  $\overline{\text{EA}}$  will be internally latched on reset.

$\overline{\text{EA}}$  should be strapped to  $V_{CC}$  for internal program executions.

This pin also receives the 12-volt programming enable voltage ( $V_{PP}$ ) during Flash programming.

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

Output from the inverting oscillator amplifier





Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 1.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

Table 1. AT89S51 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H								0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111	AUXR1 XXXXXX0				WDTRST XXXXXXXX		0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0	8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	PCON 0XXX0000	87H

AT89S51

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

**Interrupt Registers:** The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the five interrupt sources in the IP register.

Table 2. AUXR: Auxiliary Register

AUXR

Address = 8EH

Reset Value = XXX00XX0B

Not Bit Addressable

	—	—	—	WDIDLE	DISRTO	—	—	DISALE
Bit	7	6	5	4	3	2	1	0

—

Reserved for future expansion

DISALE

Disable/Enable ALE

DISALE

Operating Mode

0

ALE is emitted at a constant rate of 1/6 the oscillator frequency

1

ALE is active only during a MOVX or MOVC instruction

DISRTO

Disable/Enable Reset out

DISRTO

0

Reset pin is driven High after WDT times out

1

Reset pin is input only

WDIDLE

Disable/Enable WDT in IDLE mode

WDIDLE

0

WDT continues to count in IDLE mode

1

WDT halts counting in IDLE mode

**Dual Data Pointer Registers:** To facilitate accessing both internal and external data memory, two banks of 16-bit Data Pointer Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 in SFR AUXR1 selects DP0 and DPS = 1 selects DP1. The user should always initialize the DPS bit to the appropriate value before accessing the respective Data Pointer Register.







**Power Off Flag:** The Power Off Flag (POF) is located at bit 4 (PCON.4) in the PCON SFR. POF is set to "1" during power up. It can be set and rest under software control and is not affected by reset.

**Table 3. AUXR1: Auxiliary Register 1**

AUXR1

Address = A2H

Reset Value = XXXXXXX0B

Not Bit  
Addressable

	—	—	—	—	—	—	DPS	
Bit	7	6	5	4	3	2	1	0

— Reserved for future expansion

DPS Data Pointer Register Select

DPS

0 Selects DPTR Registers DP0L, DP0H

1 Selects DPTR Registers DP1L, DP1H

memory organization  
Program Memory

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

Data Memory

If the  $\overline{EA}$  pin is connected to GND, all program fetches are directed to external memory. On the AT89S51, if  $\overline{EA}$  is connected to  $V_{CC}$ , program fetches to addresses 0000H through FFFH are directed to internal memory and fetches to addresses 1000H through FFFFH are directed to external memory.

Watchdog timer enabled with reset-out)

The AT89S51 implements 128 bytes of on-chip RAM. The 128 bytes are accessible via direct and indirect addressing modes. Stack operations are examples of indirect addressing, so the 128 bytes of data RAM are available as stack space.

Configuring the WDT

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 14-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running. The WDT timeout period is dependent on the external clock frequency. There is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST pin.

To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, the user needs to service it by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH), and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycles. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is 98xTOSC, where TOSC=1/FOSC. To make the best use of the WDT, it



## AT89S51

should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

### WDT During Power-down and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt, which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S51 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

To ensure that the WDT does not overflow within a few states of exiting Power-down, it is best to reset the WDT just before entering Power-down mode.

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S51 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode.

With WDIDLE bit enabled, the WDT will stop to count in IDLE mode and resumes the count upon exit from IDLE.

### UART

The UART in the AT89S51 operates the same way as the UART in the AT89C51. For further information on the UART operation, refer to the ATMEL Web site (<http://www.atmel.com>). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

### Timer 0 and 1

Timer 0 and Timer 1 in the AT89S51 operate the same way as Timer 0 and Timer 1 in the AT89C51. For further information on the timers' operation, refer to the ATMEL Web site (<http://www.atmel.com>). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

### Interrupts

The AT89S51 has a total of five interrupt vectors: two external interrupts ( $\overline{INT0}$  and  $\overline{INT1}$ ), two timer interrupts (Timers 0 and 1), and the serial port interrupt. These interrupts are all shown in Figure 1.

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

Note that Table 4 shows that bit position IE.6 is unimplemented. In the AT89S51, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products.

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle.





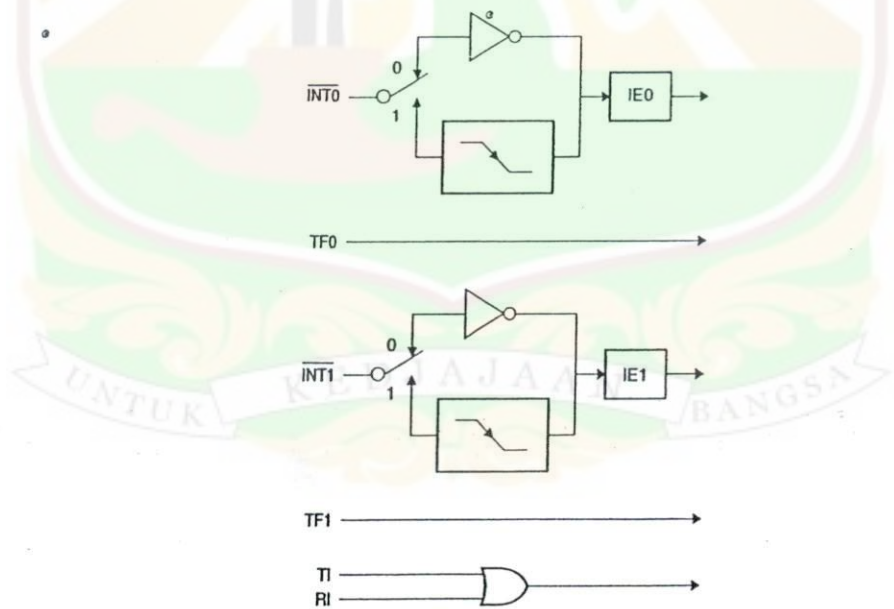
Table 4. Interrupt Enable (IE) Register

(MSB)				(LSB)			
EA	-	-	ES	ET1	EX1	ET0	EX0
Enable Bit = 1 enables the interrupt.							
Enable Bit = 0 disables the interrupt.							

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
-	IE.6	Reserved
-	IE.5	Reserved
ES	IE.4	Serial Port interrupt enable bit
ET1	IE.3	Timer 1 interrupt enable bit
EX1	IE.2	External interrupt 1 enable bit
ET0	IE.1	Timer 0 interrupt enable bit
EX0	IE.0	External interrupt 0 enable bit

User software should never write 1s to reserved bits, because they may be used in future AT89 products.

Figure 1. Interrupt Sources



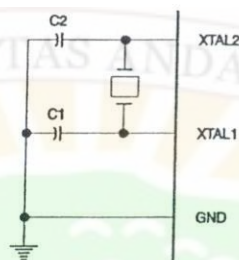


## AT89S51

Oscillator  
Characteristics

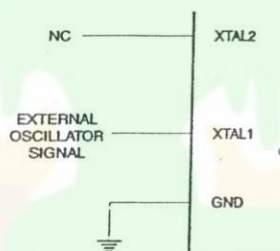
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 3. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 2. Oscillator Connections



Note: C1, C2 = 30 pF  $\pm$  10 pF for Crystals = 40 pF  $\pm$  10 pF for Ceramic Resonators

Figure 3. External Clock Drive Configuration



## Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special function registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

Note that when idle mode is terminated by a hardware reset, the device normally resumes program execution from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.

Power-down  
Mode

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by activation of an enabled external interrupt into INT0 or INT1. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before  $V_{CC}$  is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.







Table 5. Status of External Pins During Idle and Power-down Modes

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

The AT89S51 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the following table.

Table 6. Lock Bit Protection Modes

Program Lock Bits				Protection Type
	LB1	LB2	LB3	
1	U	U	U	No program lock features
2	P	U	U	MOVX instructions executed from external program memory are disabled from fetching code bytes from internal memory, EA is sampled and latched on reset, and further programming of the Flash memory is disabled
3	P	P	U	Same as mode 2, but verify is also disabled
4	P	P	P	Same as mode 3, but external execution is also disabled

When lock bit 1 is programmed, the logic level at the EA pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value and holds that value until reset is activated. The latched value of EA must agree with the current logic level at that pin in order for the device to function properly.

The AT89S51 is shipped with the on-chip Flash memory array ready to be programmed. The programming interface needs a high-voltage (12-volt) program enable signal and is compatible with conventional third-party Flash or EPROM programmers.

The AT89S51 code memory array is programmed byte-by-byte.

**Programming Algorithm:** Before programming the AT89S51, the address, data, and control signals should be set up according to the Flash programming mode table and Figures 13 and 14. To program the AT89S51, take the following steps:

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise  $\overline{EA}/V_{PP}$  to 12V.
5. Pulse ALE/PROG once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 50  $\mu$ s. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

**Data Polling:** The AT89S51 features Data Polling to indicate the end of a byte write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P0.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. Data Polling may begin any time after a write cycle has been initiated.



## AT89S51

**Ready/Busy:** The progress of byte programming can also be monitored by the  $\overline{RDY/BSY}$  output signal. P3.0 is pulled low after ALE goes high during programming to indicate  $\overline{BUSY}$ . P3.0 is pulled high again when programming is done to indicate  $\overline{READY}$ .

**Program Verify:** If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The status of the individual lock bits can be verified directly by reading them back.

**Reading the Signature Bytes:** The signature bytes are read by the same procedure as a normal verification of locations 000H, 100H, and 200H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(000H) = 1EH indicates manufactured by Atmel  
(100H) = 51H indicates 89S51  
(200H) = 06H

**Chip Erase:** In the parallel programming mode, a chip erase operation is initiated by using the proper combination of control signals and by pulsing  $\overline{ALE/PROG}$  low for a duration of 200 ns - 500 ns.

In the serial programming mode, a chip erase operation is initiated by issuing the Chip Erase instruction. In this mode, chip erase is self-timed and takes about 500 ms.

During chip erase, a serial read from any address location will return 00H at the data output.

### Programming Flash – Serial Mode

The Code memory array can be programmed using the serial ISP interface while RST is pulled to  $V_{CC}$ . The serial interface consists of pins SCK, MOSI (input) and MISO (output). After RST is set high, the Programming Enable instruction needs to be executed first before other operations can be executed. Before a reprogramming sequence can occur, a Chip Erase operation is required.

The Chip Erase operation turns the content of every memory location in the Code array into FFH.

Either an external system clock can be supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2. The maximum serial clock (SCK) frequency should be less than 1/16 of the crystal frequency. With a 33 MHz oscillator clock, the maximum SCK frequency is 2 MHz.

### Serial Programming Algorithm

To program and verify the AT89S51 in the serial programming mode, the following sequence is recommended:

1. Power-up sequence:

Apply power between VCC and GND pins.

Set RST pin to "H".

If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.

2. Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less than the CPU clock at XTAL1 divided by 16.

3. The Code array is programmed one byte at a time in either the Byte or Page mode. The write cycle is self-timed and typically takes less than 0.5 ms at 5V.

4. Any memory location can be verified by using the Read instruction that returns the content at the selected address at serial output MISO/P1.6.

5. At the end of a programming session, RST can be set low to commence normal device operation.





Power-off sequence (if needed):  
Set XTAL1 to "L" (if a crystal is not used).  
Set RST to "L".  
Turn V<sub>CC</sub> power off.

**Data Polling:** The Data Polling feature is also available in the serial mode. In this mode, during a write cycle an attempted read of the last byte written will result in the complement of the MSB of the serial output byte on MISO.

Serial  
Programming  
Instruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 8 on page 18.

Programming  
Interface –  
Parallel Mode

Every code byte in the Flash array can be programmed by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Table 7. Flash Programming Modes

Mode	V <sub>CC</sub>	RST	PSEN	ALE/ PROG	EA/ V <sub>PP</sub>	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.3-0	P1.7-0
												Address	
Write Code Data	5V	H	L	(2)	12V	L	H	H	H	H	D <sub>IN</sub>	A11-8	A7-0
Read Code Data	5V	H	L	H	H	L	L	L	H	H	D <sub>OUT</sub>	A11-8	A7-0
Write Lock Bit 1	5V	H	L	(3)	12V	H	H	H	H	H	X	X	X
Write Lock Bit 2	5V	H	L	(3)	12V	H	H	H	L	L	X	X	X
Write Lock Bit 3	5V	H	L	(3)	12V	H	L	H	H	L	X	X	X
Read Lock Bits 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Chip Erase	5V	H	L	(1)	12V	H	L	H	L	L	X	X	X
Read Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	0000	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	51H	0001	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	06H	0010	00H

- Notes:
1. Each PROG pulse is 200 ns - 500 ns for Chip Erase.
  2. Each PROG pulse is 200 ns - 500 ns for Write Code Data.
  3. Each PROG pulse is 200 ns - 500 ns for Write Lock Bits.
  4. RDY/BSY signal is output on P3.0 during programming.
  5. X = don't care.



## AT89S51

Figure 4. Programming the Flash Memory (Parallel Mode)

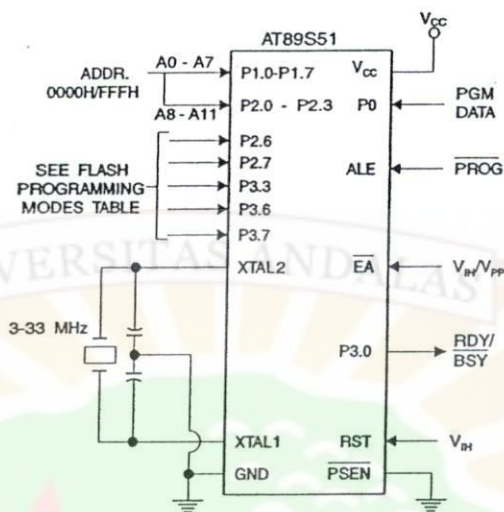
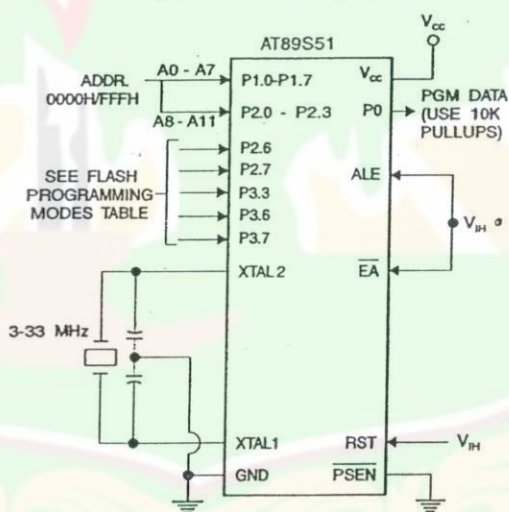


Figure 5. Verifying the Flash Memory (Parallel Mode)



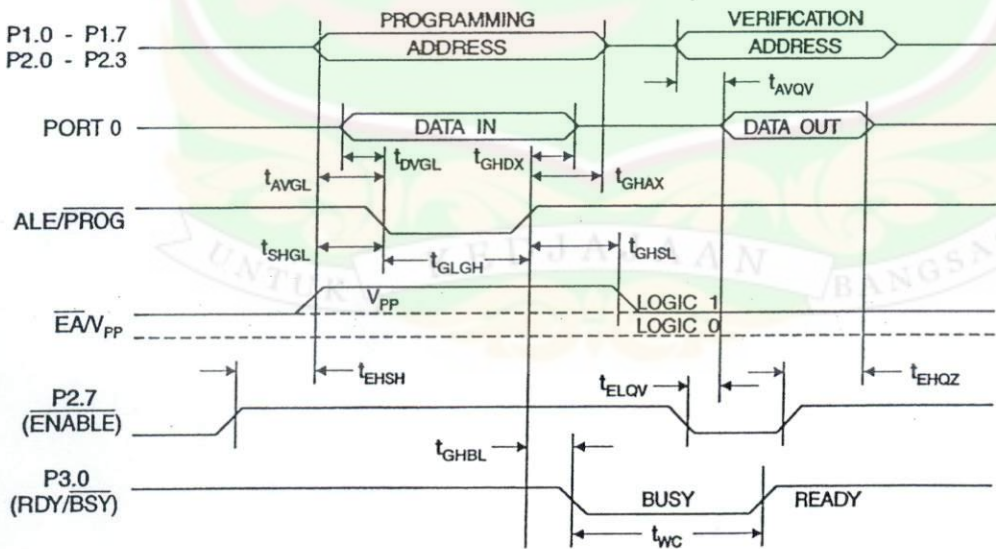


Flash Programming and Verification Characteristics (Parallel Mode)

T<sub>a</sub> = 20°C to 30°C, V<sub>CC</sub> = 4.5 to 5.5V

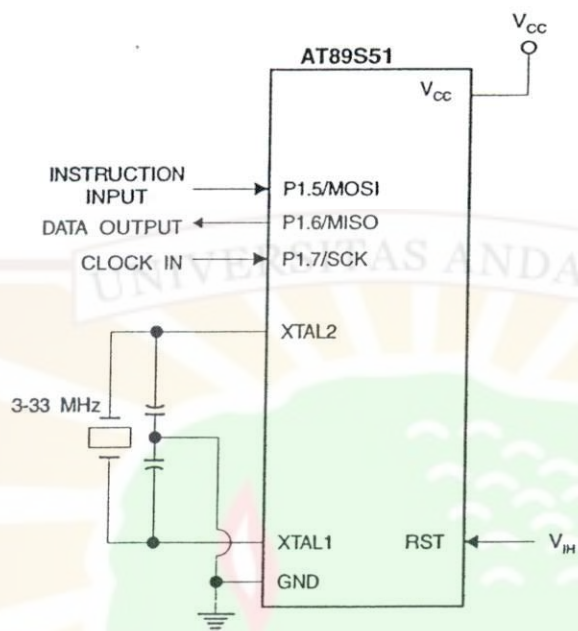
Symbol	Parameter	Min	Max	Units
V <sub>PP</sub>	Programming Supply Voltage	11.5	12.5	V
I <sub>PP</sub>	Programming Supply Current		10	mA
I <sub>CC</sub>	V <sub>CC</sub> Supply Current		30	mA
f <sub>CLCL</sub>	Oscillator Frequency	3	33	MHz
t <sub>AVGL</sub>	Address Setup to $\overline{\text{PROG}}$ Low	48t <sub>CLCL</sub>		
t <sub>GHAX</sub>	Address Hold After $\overline{\text{PROG}}$	48t <sub>CLCL</sub>		
t <sub>AVGL</sub>	Data Setup to $\overline{\text{PROG}}$ Low	48t <sub>CLCL</sub>		
t <sub>GHDX</sub>	Data Hold After $\overline{\text{PROG}}$	48t <sub>CLCL</sub>		
t <sub>SH</sub>	P2.7 ( $\overline{\text{ENABLE}}$ ) High to V <sub>PP</sub>	48t <sub>CLCL</sub>		
t <sub>AVGL</sub>	V <sub>PP</sub> Setup to $\overline{\text{PROG}}$ Low	10		μs
t <sub>GHSL</sub>	V <sub>PP</sub> Hold After $\overline{\text{PROG}}$	10		μs
t <sub>GLGH</sub>	$\overline{\text{PROG}}$ Width	0.2	1	μs
t <sub>AVQV</sub>	Address to Data Valid		48t <sub>CLCL</sub>	
t <sub>AVQV</sub>	$\overline{\text{ENABLE}}$ Low to Data Valid		48t <sub>CLCL</sub>	
t <sub>ELQV</sub>	Data Float After $\overline{\text{ENABLE}}$	0	48t <sub>CLCL</sub>	
t <sub>GHBL</sub>	$\overline{\text{PROG}}$ High to $\overline{\text{BUSY}}$ Low		1.0	μs
t <sub>WC</sub>	Byte Write Cycle Time		50	μs

Figure 6. Flash Programming and Verification Waveforms – Parallel Mode



# AT89S51

Figure 7. Flash Memory Serial Downloading



## Flash Programming and Verification Waveforms – Serial Mode

Figure 8. Serial Programming Waveforms

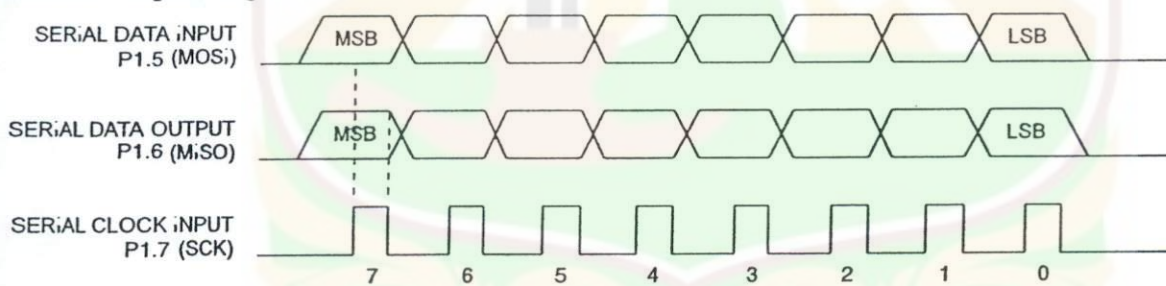






Table 8. Serial Programming Instruction Set

Instruction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte 4	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output)	Enable Serial Programming while RST is high
Chip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
Read Program Memory (Byte Mode)	0010 0000	xxxx			Read data from Program memory in the byte mode
Write Program Memory (Byte Mode)	0100 0000	xxxx			Write data to Program memory in the byte mode
Write Lock Bits <sup>(2)</sup>	1010 1100	1110 00	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (2).
Read Lock Bits	0010 0100	xxxx xxxx	xxxx xxxx		Read back current status of the lock bits (a programmed lock bit reads back as a "1")
Read Signature Bytes <sup>(1)</sup>	0010 1000	xxx		Signature Byte	Read Signature Byte
Read Program Memory (Page Mode)	0011 0000	xxxx	Byte 0	Byte 1... Byte 255	Read data from Program memory in the Page Mode (256 bytes)
Write Program Memory (Page Mode)	0101 0000	xxxx	Byte 0	Byte 1... Byte 255	Write data to Program memory in the Page Mode (256 bytes)

1. The signature bytes are not readable in Lock Bit Modes 3 and 4.

2. B1 = 0, B2 = 0 → Mode 1, no lock protection  
B1 = 0, B2 = 1 → Mode 2, lock bit 1 activated  
B1 = 1, B2 = 0 → Mode 3, lock bit 2 activated  
B1 = 1, B2 = 1 → Mode 4, lock bit 3 activated

}

Each of the lock bits needs to be activated sequentially before Mode 4 can be executed.

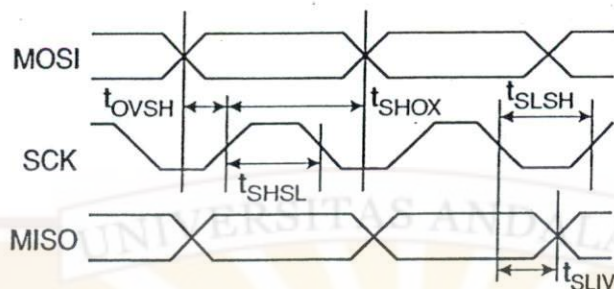
After Reset signal is high, SCK should be low for at least 64 system clocks before it goes high to clock in the enable data bytes. No pulsing of Reset signal is necessary. SCK should be no faster than 1/16 of the system clock at XTAL1.

For Page Read/Write, the data always starts from byte 0 to 255. After the command byte and upper address byte are latched, each byte thereafter is treated as data until all 256 bytes are shifted in/out. Then the next instruction will be ready to be decoded.

## AT89S51

## Serial Programming Characteristics

Figure 9. Serial Programming Timing

Table 9. Serial Programming Characteristics,  $T_A = -40^\circ\text{C}$  to  $85^\circ\text{C}$ ,  $V_{CC} = 4.0 - 5.5\text{V}$  (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0		33	MHz
$t_{CLCL}$	Oscillator Period	30			ns
$t_{SHSL}$	SCK Pulse Width High	$8 t_{CLCL}$			ns
$t_{SLSH}$	SCK Pulse Width Low	$8 t_{CLCL}$			ns
$t_{OVSH}$	MOSI Setup to SCK High	$t_{CLCL}$			ns
$t_{SHOX}$	MOSI Hold after SCK High	$2 t_{CLCL}$			ns
$t_{SLIV}$	SCK Low to MISO Valid	10	16	32	ns
ERASE	Chip Erase Instruction Cycle Time			500	ms
SWC	Serial Byte Write Cycle Time			$64 t_{CLCL} + 400$	$\mu\text{s}$





Absolute Maximum Ratings\*

Operating Temperature.....	-55°C to +125°C
Storage Temperature .....	-65°C to +150°C
Voltage on Any Pin With Respect to Ground .....	-1.0V to +7.0V
Maximum Operating Voltage .....	6.6V
IO Output Current.....	15.0 mA

\*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Electrical Characteristics

The values shown in this table are valid for TA = -40°C to 85°C and VCC = 4.0V to 5.5V, unless otherwise noted.

Symbol	Parameter	Condition	Min	Max	Units
IL	Input Low Voltage	(Except EA)	-0.5	0.2 VCC-0.1	V
ILI	Input Low Voltage (EA)		-0.5	0.2 VCC-0.3	V
IHI	Input High Voltage	(Except XTAL1, RST)	0.2 VCC+0.9	VCC+0.5	V
IHI	Input High Voltage	(XTAL1, RST)	0.7 VCC	VCC+0.5	V
OL	Output Low Voltage <sup>(1)</sup> (Ports 1,2,3)	IOL = 1.6 mA		0.45	V
OLI	Output Low Voltage <sup>(1)</sup> (Port 0, ALE, PSEN)	IOL = 3.2 mA		0.45	V
OH	Output High Voltage (Ports 1,2,3, ALE, PSEN)	I <sub>OH</sub> = -60 µA, VCC = 5V ± 10%	2.4		V
		I <sub>OH</sub> = -25 µA	0.75 VCC		V
		I <sub>OH</sub> = -10 µA	0.9 VCC		V
OH	Output High Voltage (Port 0 in External Bus Mode)	I <sub>OH</sub> = -800 µA, VCC = 5V ± 10%	2.4		V
		I <sub>OH</sub> = -300 µA	0.75 VCC		V
		I <sub>OH</sub> = -80 µA	0.9 VCC		V
	Logical 0 Input Current (Ports 1,2,3)	VIN = 0.45V		-50	µA
	Logical 1 to 0 Transition Current (Ports 1,2,3)	VIN = 2V, VCC = 5V ± 10%		-650	µA
	Input Leakage Current (Port 0, EA)	0.45 < VIN < VCC		±10	µA
RST	Reset Pulldown Resistor		50	300	KΩ
	Pin Capacitance	Test Freq. = 1 MHz, TA = 25°C		10	pF
	Power Supply Current	Active Mode, 12 MHz		25	mA
		Idle Mode, 12 MHz		6.5	mA
	Power-down Mode <sup>(2)</sup>	VCC = 5.5V		50	µA

1. Under steady state (non-transient) conditions, IOL must be externally limited as follows:  
Maximum IOL per port pin: 10 mA  
Maximum IOL per 8-bit port:  
Port 0: 26 mA      Ports 1, 2, 3: 15 mA  
Maximum total IOL for all output pins: 71 mA  
If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum VCC for Power-down is 2V.



AT89S51

C Characteristics

der operating conditions, load capacitance for Port 0, ALE/PROG, and PSEN = 100 pF; load capacitance for all other  
puts = 80 pF.

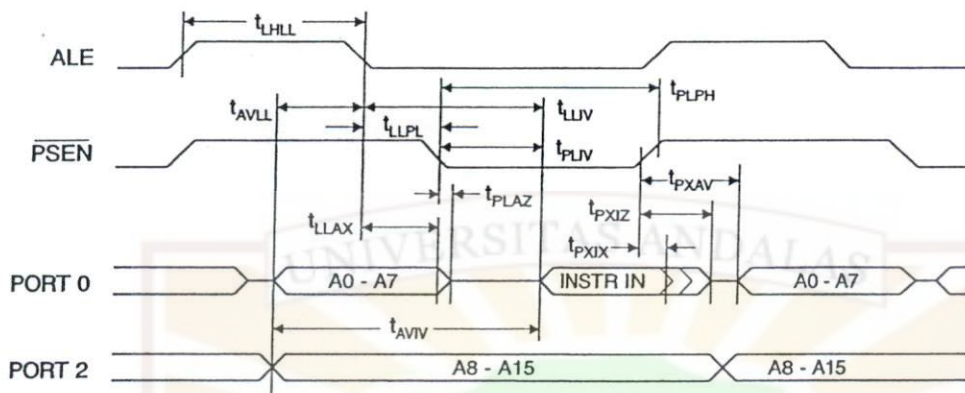
External Program and Data Memory Characteristics

Symbol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
f <sub>CLCL</sub>	Oscillator Frequency			0	33	MHz
t <sub>ILL</sub>	ALE Pulse Width	127		2t <sub>CLCL</sub> -40		ns
t <sub>LL</sub>	Address Valid to ALE Low	43		t <sub>CLCL</sub> -25		ns
t <sub>MAX</sub>	Address Hold After ALE Low	48		t <sub>CLCL</sub> -25		ns
t <sub>IV</sub>	ALE Low to Valid Instruction In		233		4t <sub>CLCL</sub> -65	ns
t <sub>PL</sub>	ALE Low to PSEN Low	43		t <sub>CLCL</sub> -25		ns
t <sub>PH</sub>	PSEN Pulse Width	205		3t <sub>CLCL</sub> -45		ns
t <sub>IV</sub>	PSEN Low to Valid Instruction In		145		3t <sub>CLCL</sub> -60	ns
t <sub>IX</sub>	Input Instruction Hold After PSEN	0		0		ns
t <sub>IZ</sub>	Input Instruction Float After PSEN		59		t <sub>CLCL</sub> -25	ns
t <sub>AV</sub>	PSEN to Address Valid	75		t <sub>CLCL</sub> -8		ns
t <sub>V</sub>	Address to Valid Instruction In		312		5t <sub>CLCL</sub> -80	ns
t <sub>IZ</sub>	PSEN Low to Address Float		10		10	ns
t <sub>RH</sub>	RD Pulse Width	400		6t <sub>CLCL</sub> -100		ns
t <sub>WH</sub>	WR Pulse Width	400		6t <sub>CLCL</sub> -100		ns
t <sub>V</sub>	RD Low to Valid Data In		252		5t <sub>CLCL</sub> -90	ns
t <sub>IX</sub>	Data Hold After RD	0		0		ns
t <sub>IZ</sub>	Data Float After RD		97		2t <sub>CLCL</sub> -28	ns
t <sub>V</sub>	ALE Low to Valid Data In		517		8t <sub>CLCL</sub> -150	ns
t <sub>V</sub>	Address to Valid Data In		585		9t <sub>CLCL</sub> -165	ns
t <sub>L</sub>	ALE Low to RD or WR Low	200	300	3t <sub>CLCL</sub> -50	3t <sub>CLCL</sub> +50	ns
t <sub>L</sub>	Address to RD or WR Low	203		4t <sub>CLCL</sub> -75		ns
t <sub>IX</sub>	Data Valid to WR Transition	23		t <sub>CLCL</sub> -30		ns
t <sub>WH</sub>	Data Valid to WR High	433		7t <sub>CLCL</sub> -130		ns
t <sub>IX</sub>	Data Hold After WR	33		t <sub>CLCL</sub> -25		ns
t <sub>IZ</sub>	RD Low to Address Float		0		0	ns
t <sub>H</sub>	RD or WR High to ALE High	43	123	t <sub>CLCL</sub> -25	t <sub>CLCL</sub> +25	ns

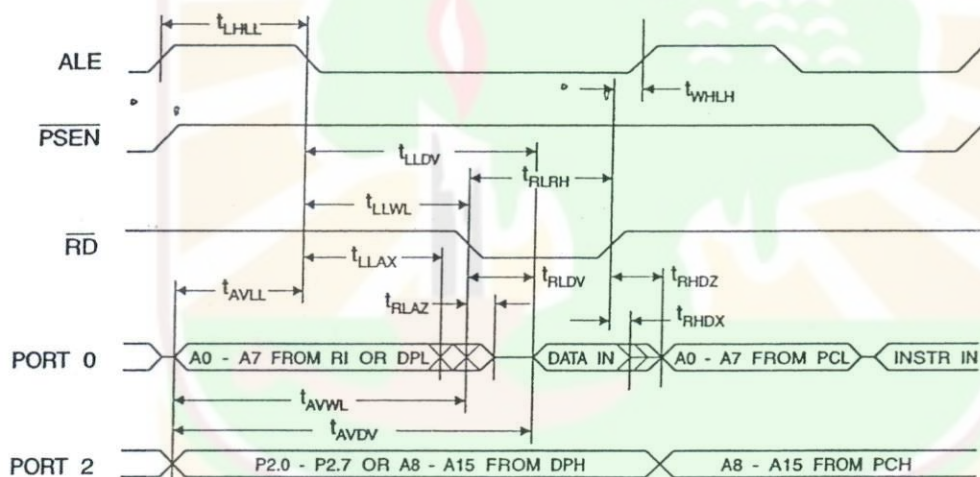




## External Program Memory Read Cycle

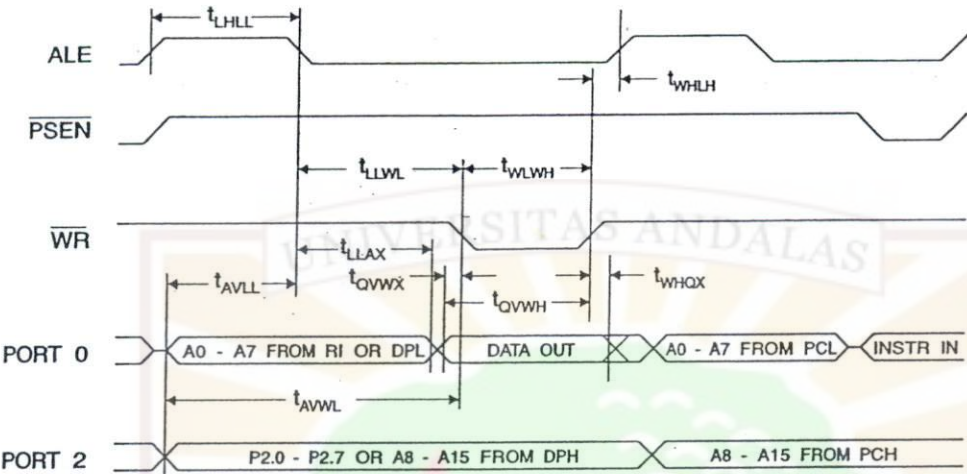


## External Data Memory Read Cycle

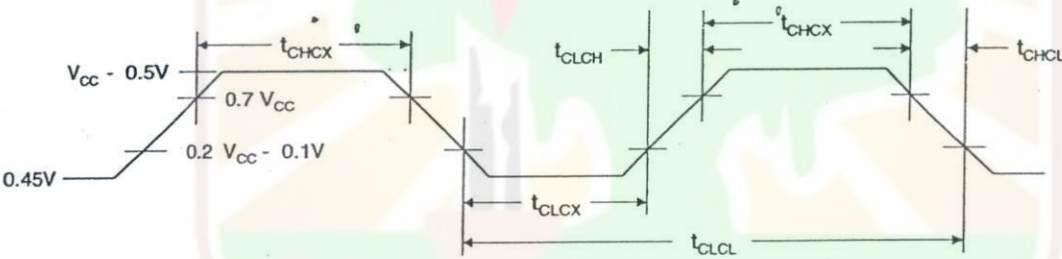


AT89S51

Internal Data Memory Write Cycle



Internal Clock Drive Waveforms



Internal Clock Drive

Symbol	Parameter	Min	Max	Units
$f_{osc}$	Oscillator Frequency	0	33	MHz
$T$	Clock Period	30		ns
$t_{CH}$	High Time	12		ns
$t_{CL}$	Low Time	12		ns
$t_{R}$	Rise Time		5	ns
$t_{F}$	Fall Time		5	ns





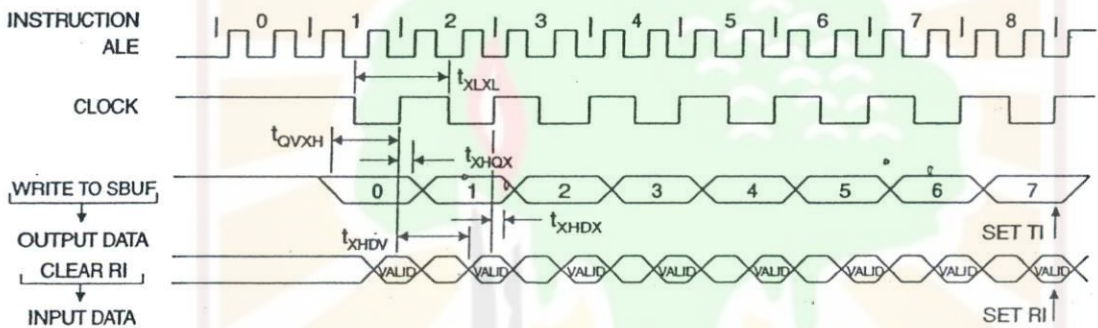


Serial Port Timing: Shift Register Mode Test Conditions

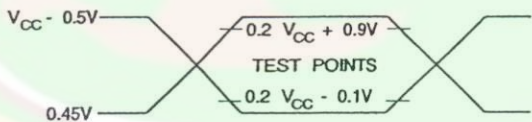
The values in this table are valid for  $V_{CC} = 4.0V$  to  $5.5V$  and Load Capacitance =  $80\text{ pF}$ .

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
$t_{XLXL}$	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		$\mu s$
$t_{QVXH}$	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}-133$		ns
$t_{XHGX}$	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}-80$		ns
$t_{XHDX}$	Input Data Hold After Clock Rising Edge	0		0		ns
$t_{XHDV}$	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}-133$	ns

Shift Register Mode Timing Waveforms

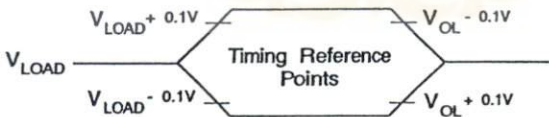


AC Testing Input/Output Waveforms<sup>(1)</sup>



1. AC Inputs during testing are driven at  $V_{CC} - 0.5V$  for a logic 1 and  $0.45V$  for a logic 0. Timing measurements are made at  $V_{IH}$  min. for a logic 1 and  $V_{IL}$  max. for a logic 0.

Float Waveforms<sup>(1)</sup>



1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded  $V_{OH}/V_{OL}$  level occurs.

AT89S51

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC	44A	Commercial (0°C to 70°C)
		AT89S51-24JC	44J	
		AT89S51-24PC	40P6	
		AT89S51-24AI	44A	Industrial (-40°C to 85°C)
		AT89S51-24JI	44J	
		AT89S51-24PI	40P6	
33	4.5V to 5.5V	AT89S51-33AC	44A	Commercial (0°C to 70°C)
		AT89S51-33JC	44J	
		AT89S51-33PC	40P6	

 = Preliminary Availability



Package Type	
	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
	44-lead, Plastic J-leaded Chip Carrier (PLCC)
	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)

